Summary Lecture Requirements Engineering

Lecture 11, DAT230, Requirements Engineering
Robert Feldt, 2012-10-17

Group assignment

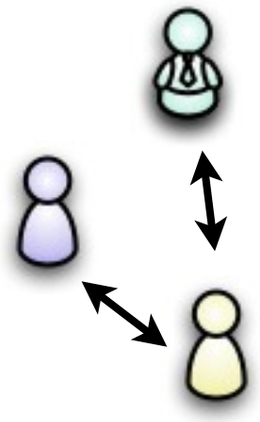
- "Problematic" groups and group members will be further investigated
- We know of 1 problematic group so far
  - Little to no effort on group assignment
  - Likely to be many more; report or forget
  - We will contact problematic ones in coming weeks
- If we judge that you have not contributed enough
  - No point bonus/"cushion" on written exam (even retroactively)
  - Fail group assignment - rework

Material for written exam

- All chapters for the book listed on home page
- All articles linked to from “Schedule” page on course web
- All lecture and WS slides/material
- Assignment 1&3 material and what you learnt from there

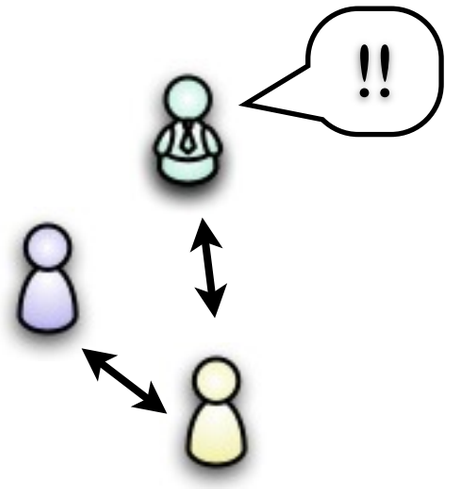
Document

Stakeholders



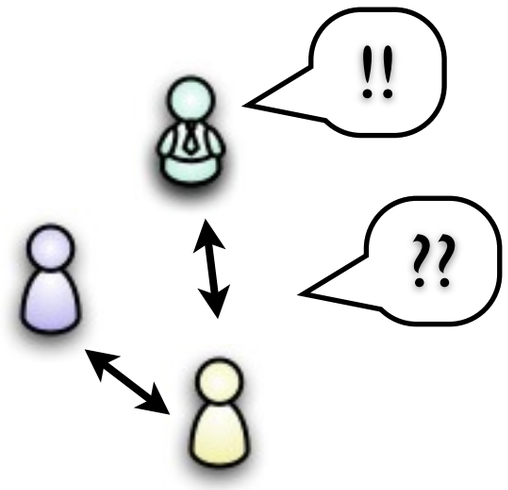
Relations

Say

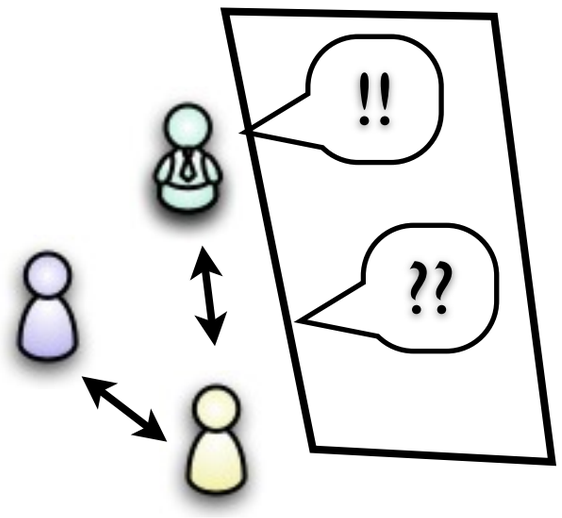


Need!

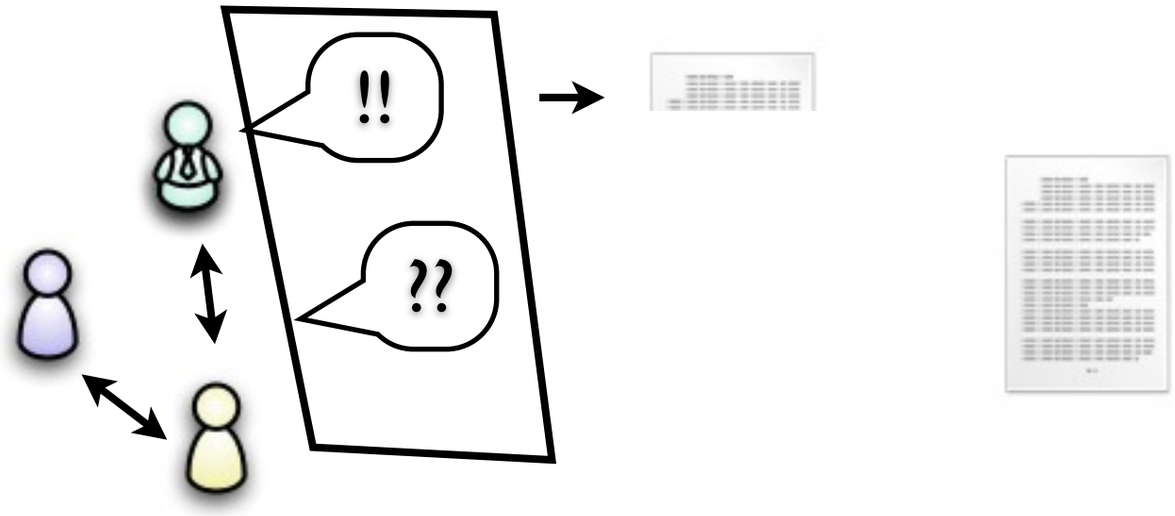
Say Think



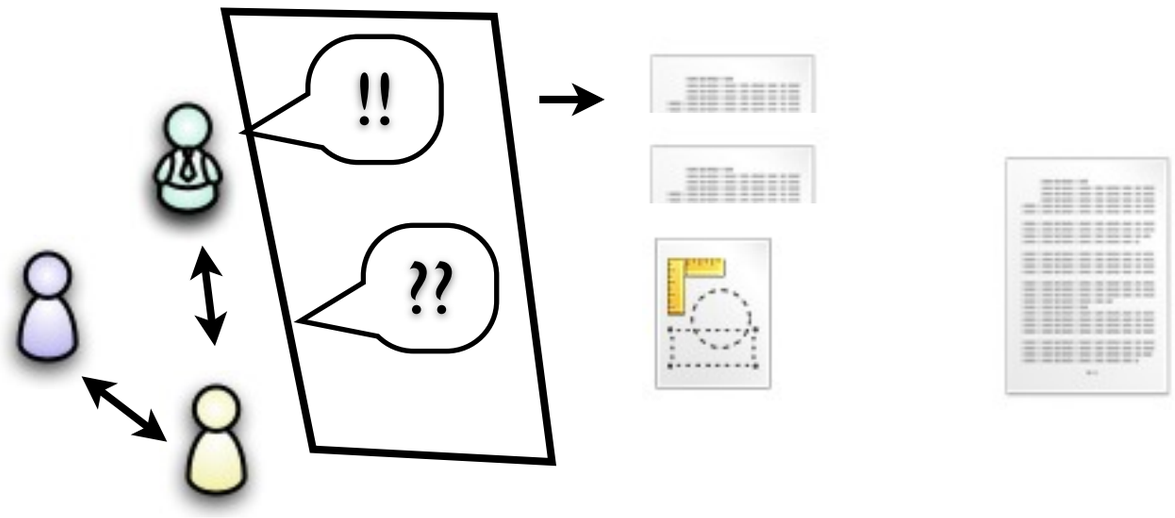
Capture

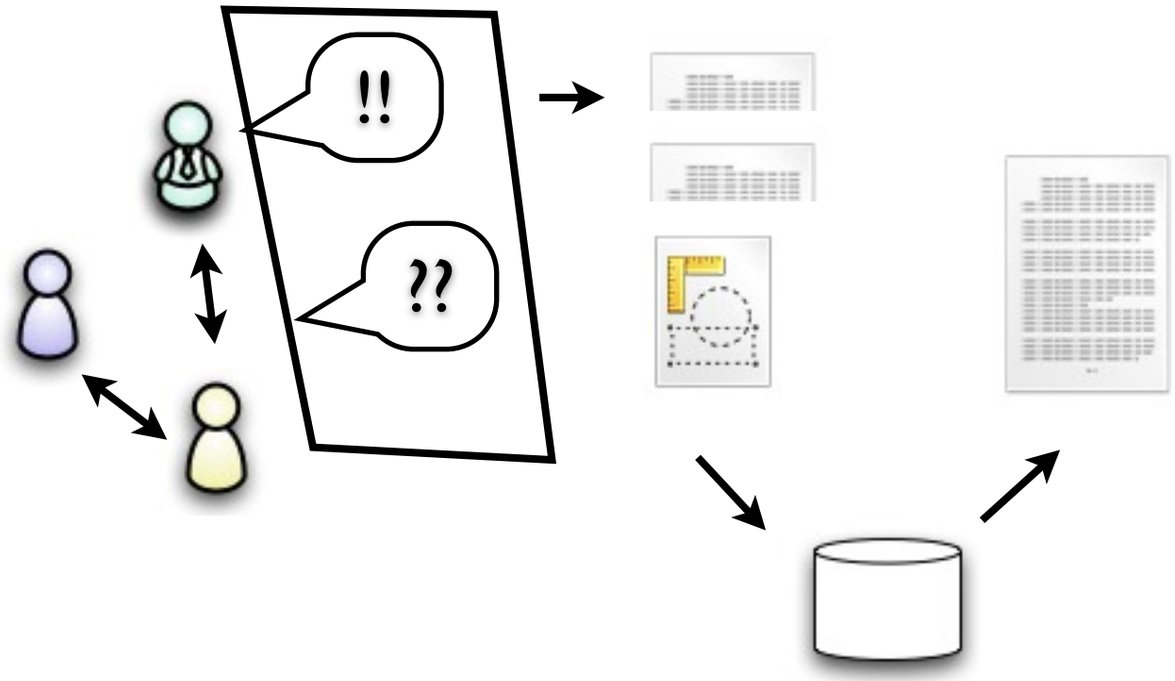


Transform

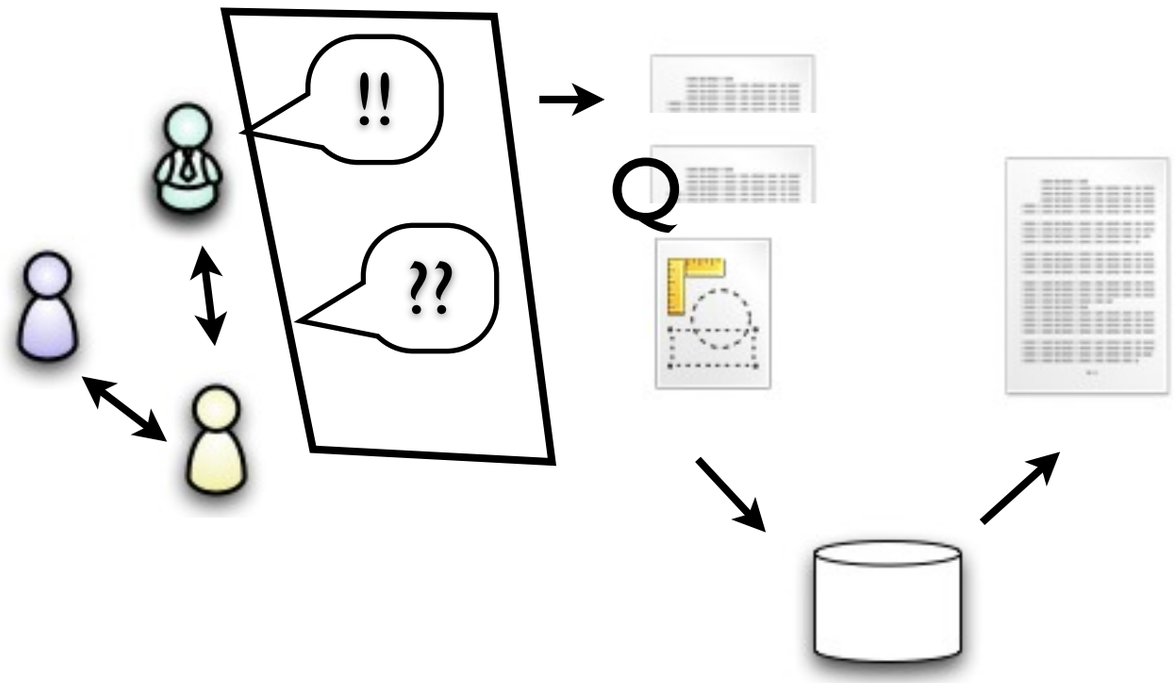


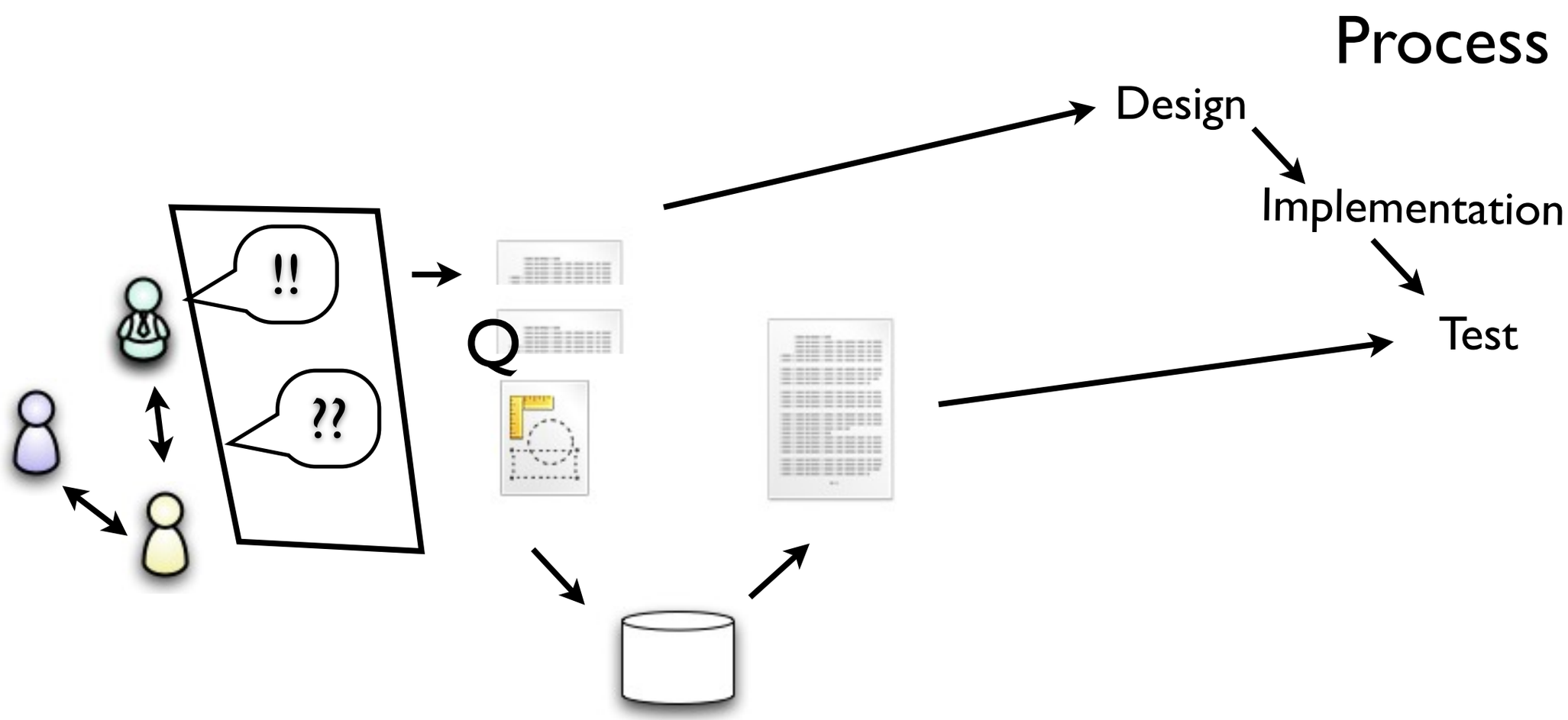
Specify

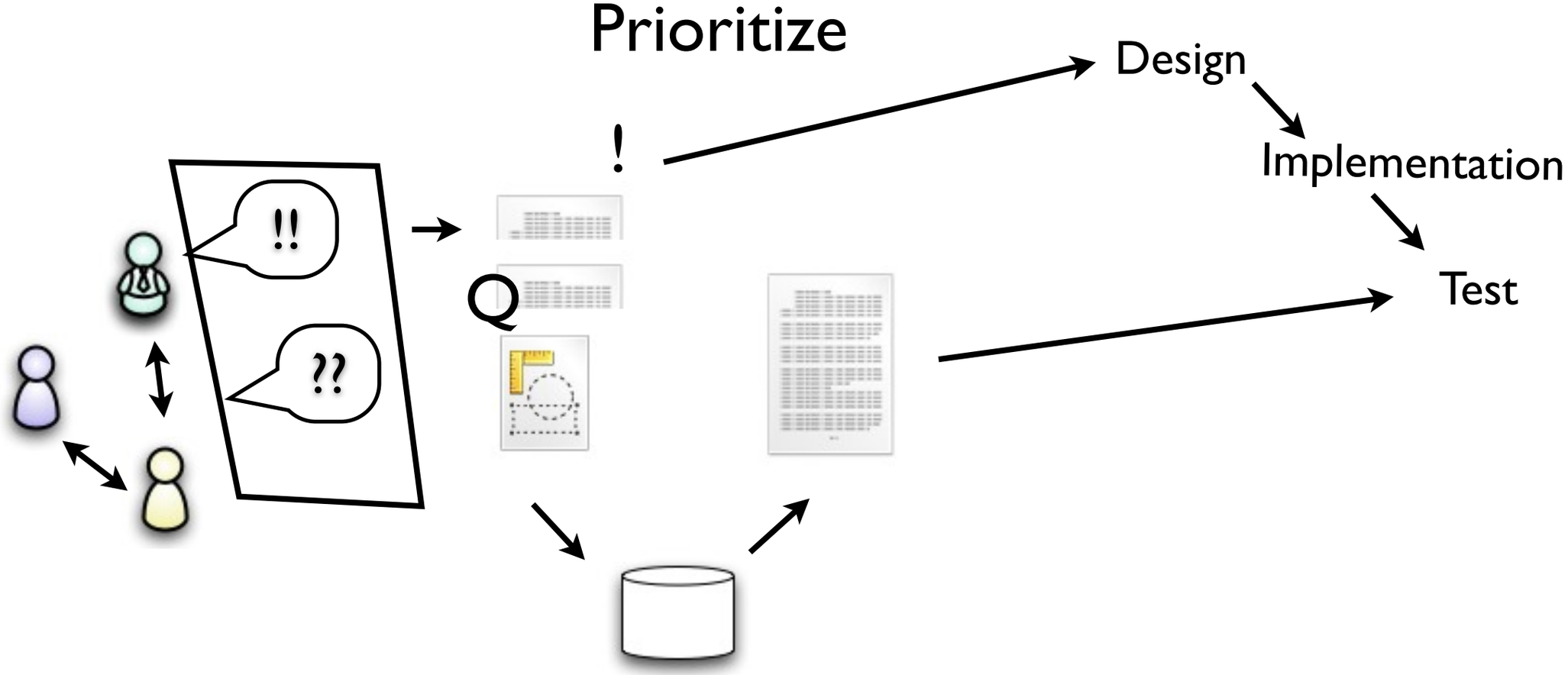


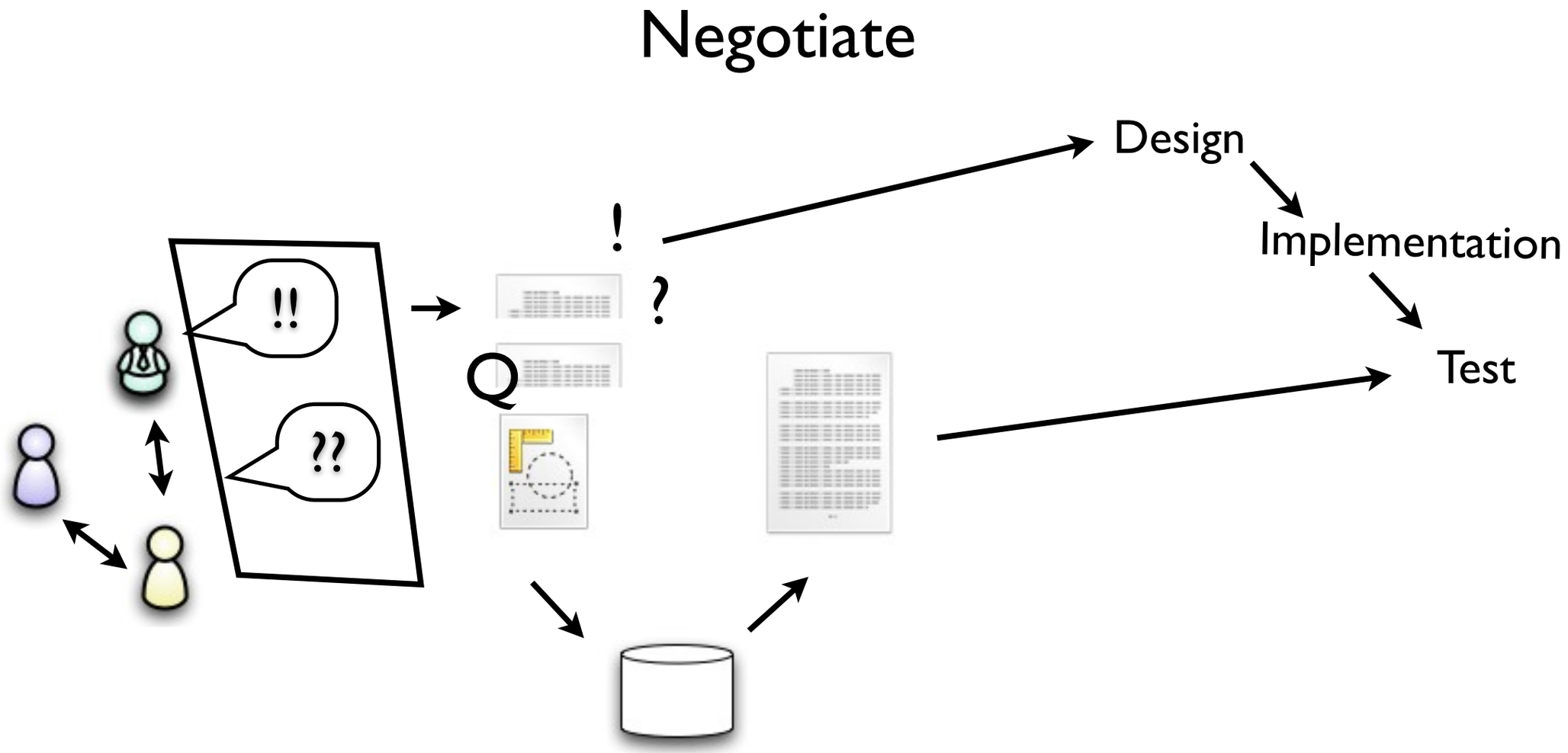

!!
??
Store

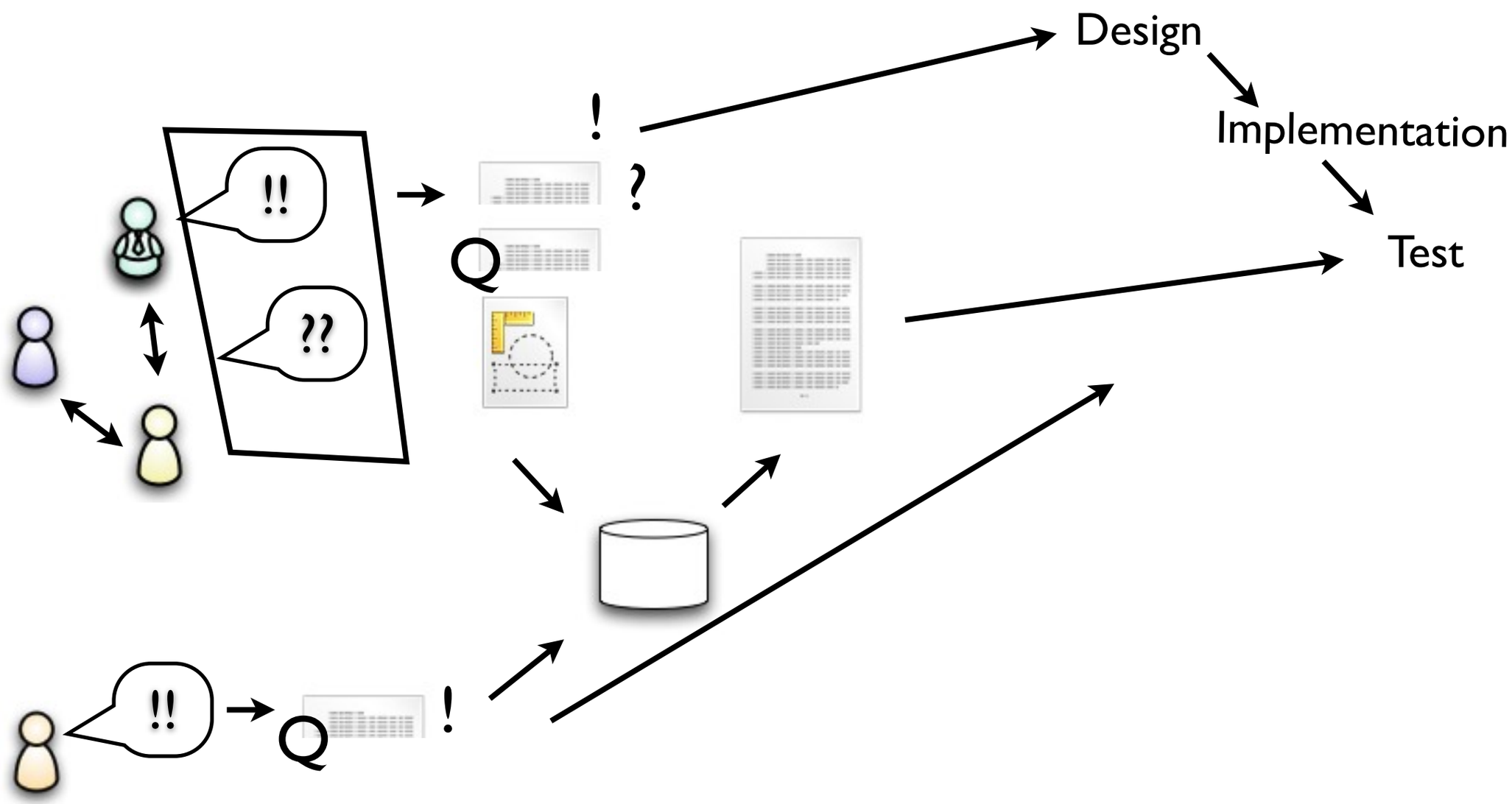
Validation

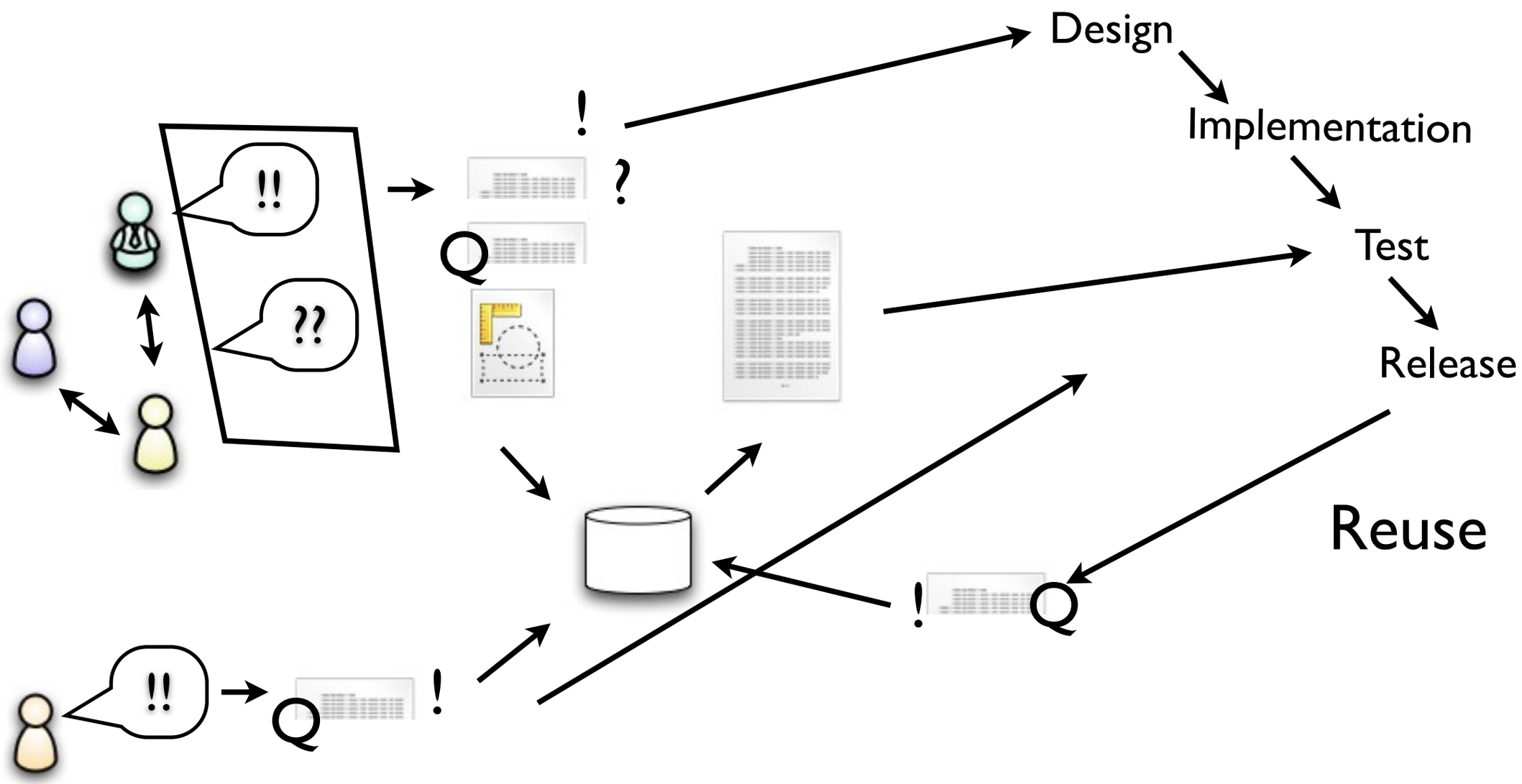



Process
Design
Implementation
Test
!!
??
Q

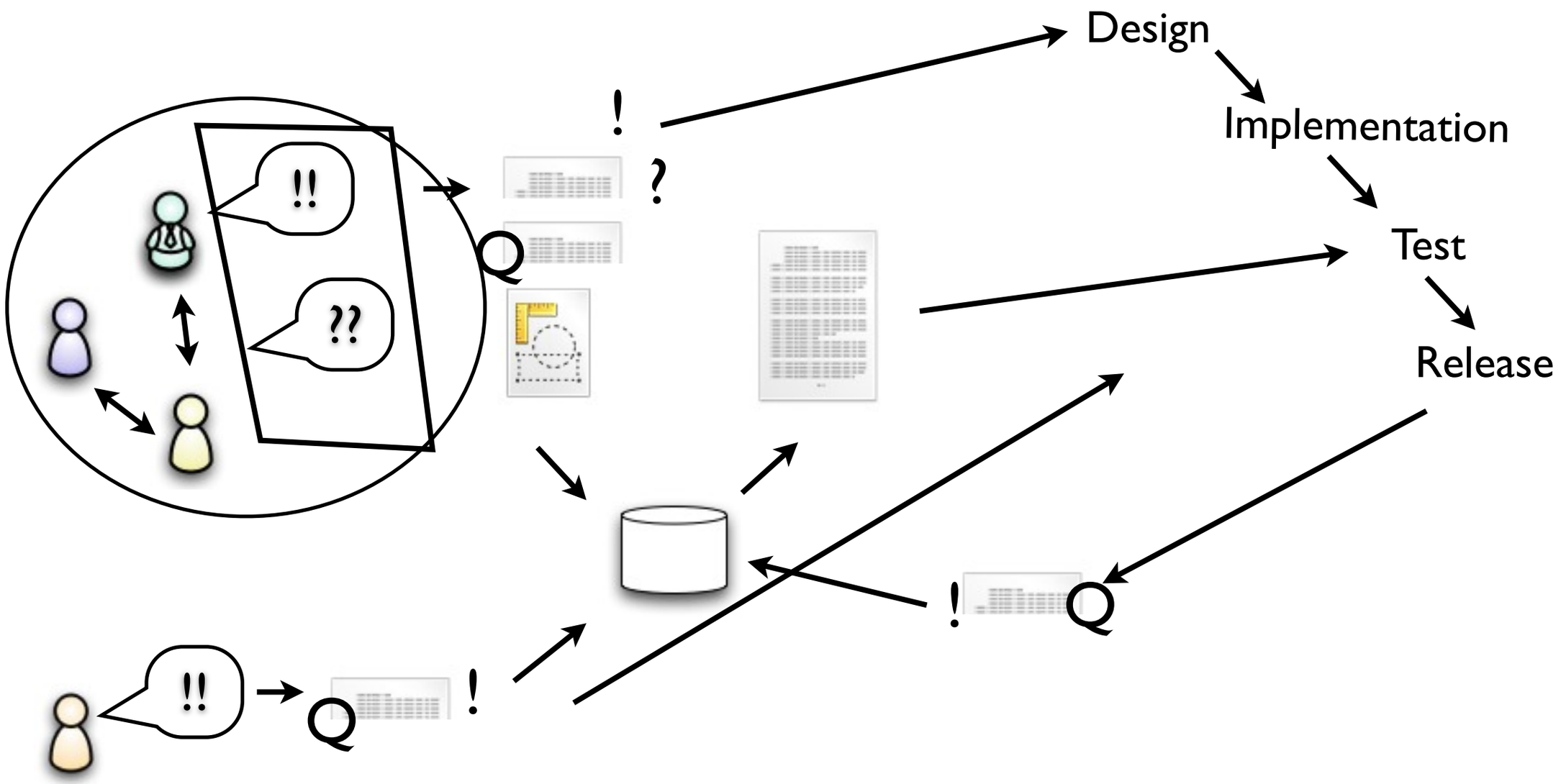

Prioritize
!
Design
Implementation
Test
!!
??
Q


Negotiate
!!
??
!
?
Q
Design
Implementation
Test

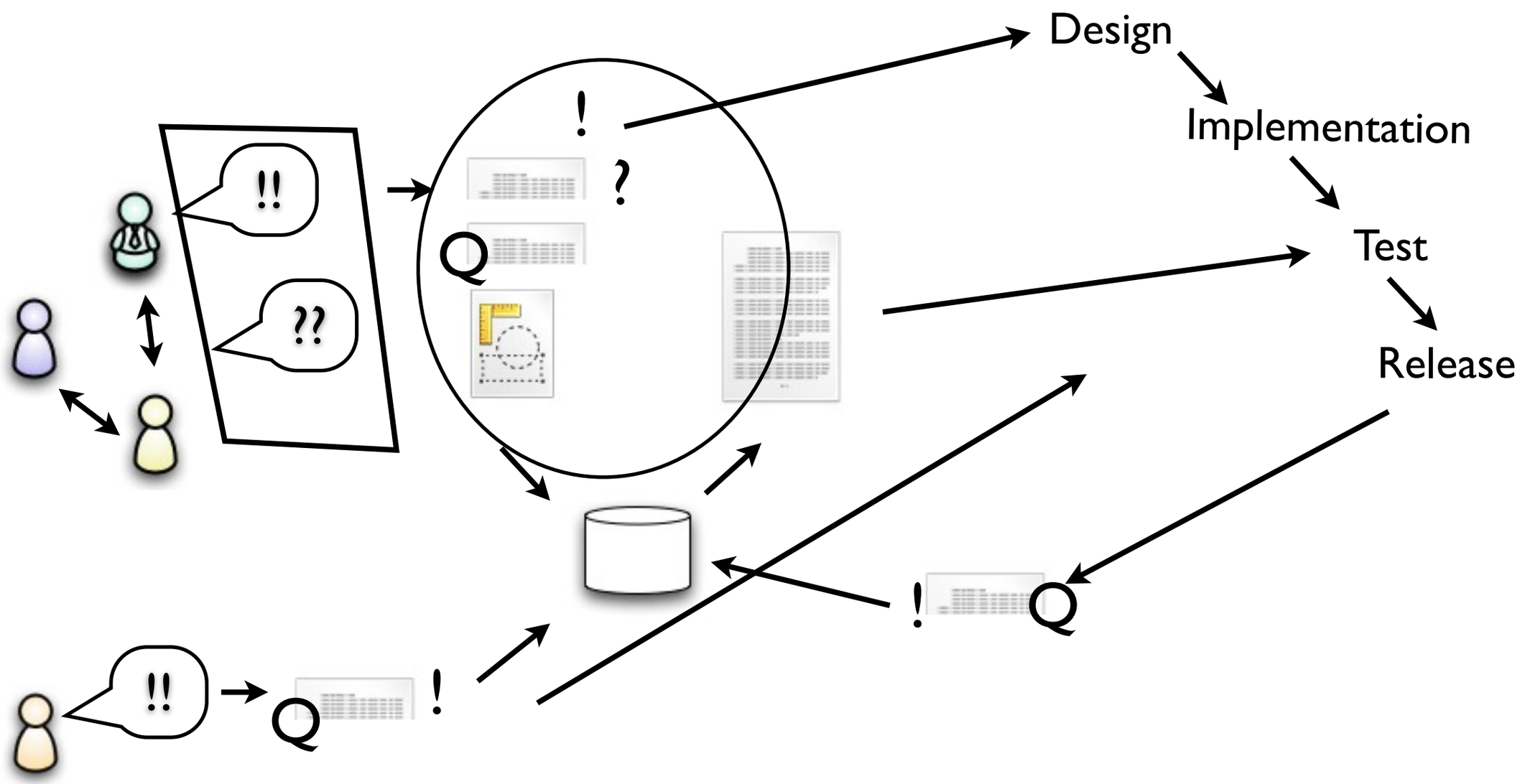

Design
Implementation
Test
!
?
Q
!!
??
!!
Q
!
Change

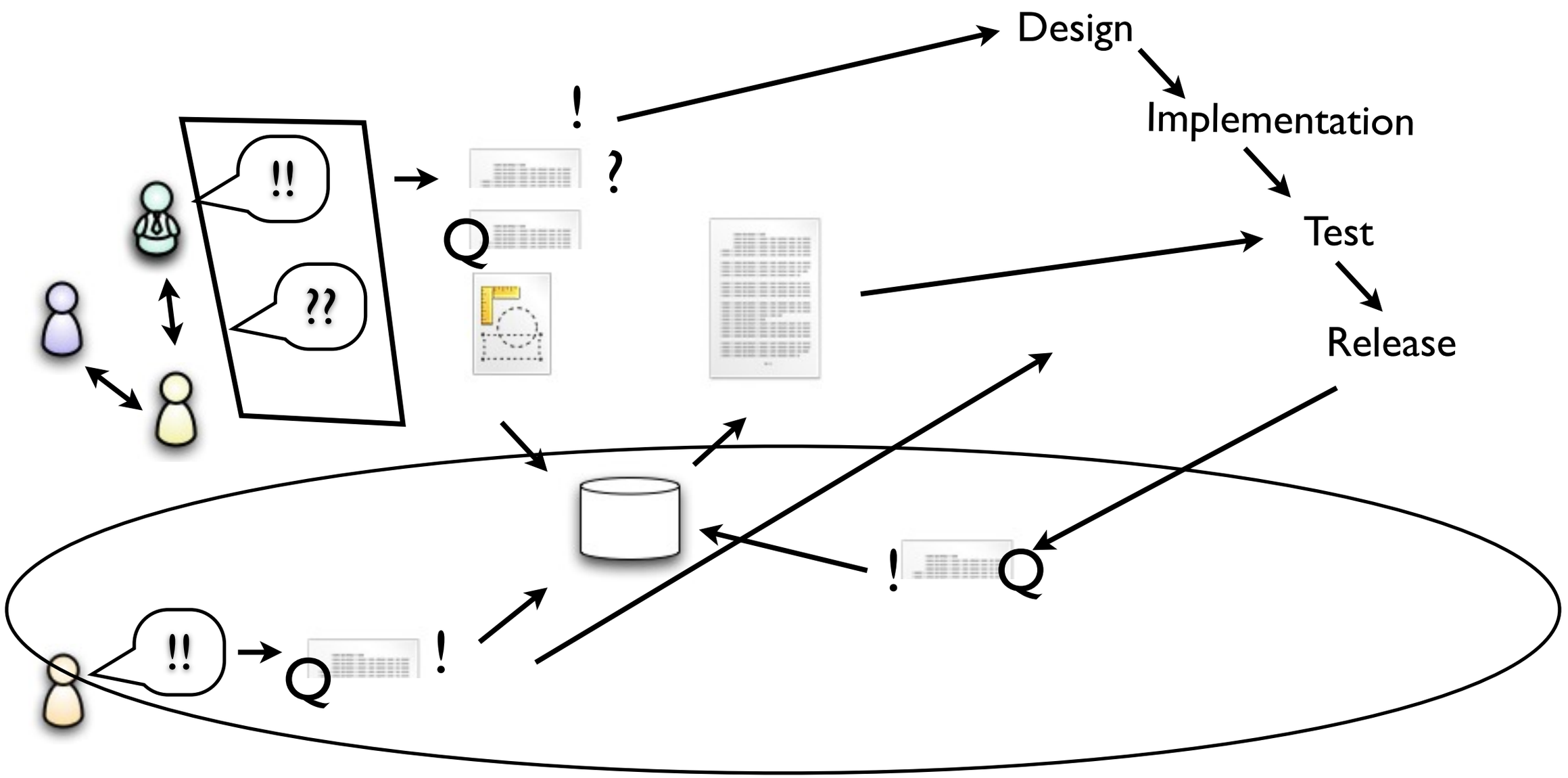

!!
??
!
?
Q
Design
Implementation
Test
Release
Reuse
!!
Q
!
!
Q

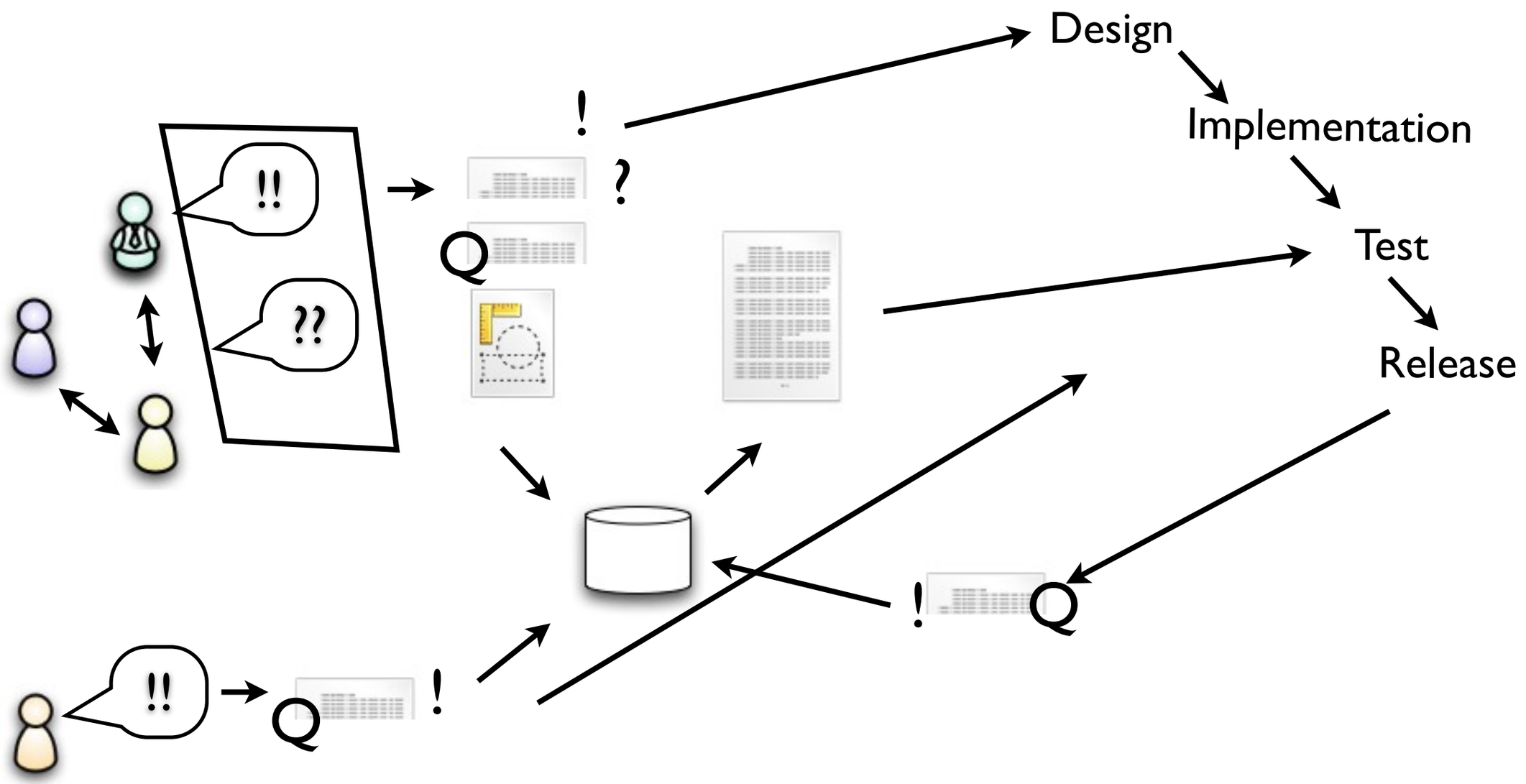
Elicitation



Specification & Analysis




Design
Implementation
Test
Release
!
?
Q
!!
??
!!
Q
!
!
Q
Management


Design
Implementation
Test
Release
!
?
Q
!!
??
!!
Q
!
!
Q

Elicitation

- "understand customers'/users' view of their problems/ opportunities"
  - Understand enough to proceed with _____ ____
  - Never think ___ understand better than ___
  - Never assume one _____ can speak for all _____
  - Maintain a _____ of terms
  - Prepare for ____ even after elicitation
  - Stakeholders have the right to _____ their mind
  - Combine multiple _____ to _____ results

Elicitation methods

Explicit

Interviews
Questionnaires
Doc analysis
Archaeology

"Traditional"/
Survey

Reflective

Think-aloud /
Protocol Analysis
Laddering
Card sorting
Repertory grids

"Cognitive"/
Introspective

Creativity

Brainstorming
Focus groups
JAD/RAD
Req Workshops

Consensus

Group-based

Implicit

Ethnography
Observation
Apprenticing
Conversation analysis

Contextual/
Observation

Working prototypes
Mashups
Drawings
Diagramming

Reactive

Refining

Prototyping

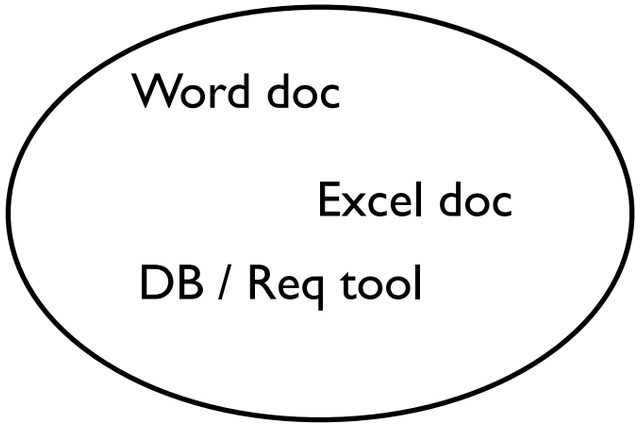
I*
KAOS
CREWS
Use Cases

Model- or
Spec-driven

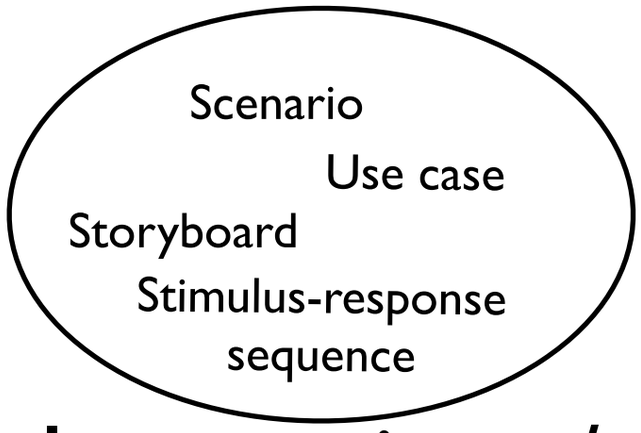
Specification

- "record understandings so all parties see what to expect from system"
  - goal is to spec to enough _____ so different stakeholders are ____ in their interpretation
  - select spec notations that customers _____
  - construct _____ where nat lang introduces high risk
  - use right ____ for the right job
  - customers want their problem solved, not to learn new _____

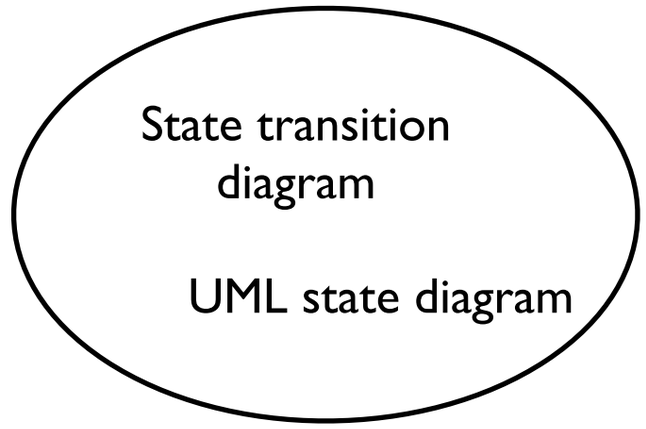
Specification Techniques



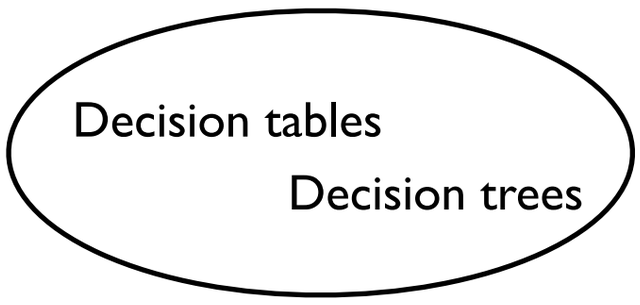
Text



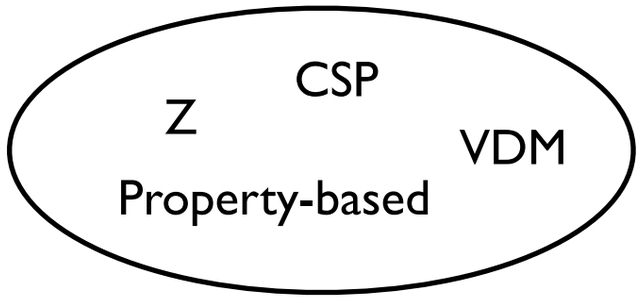
Interaction- / Sequence-based



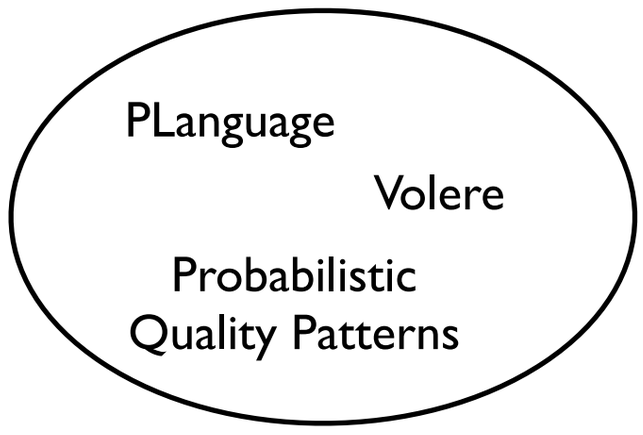
State-based



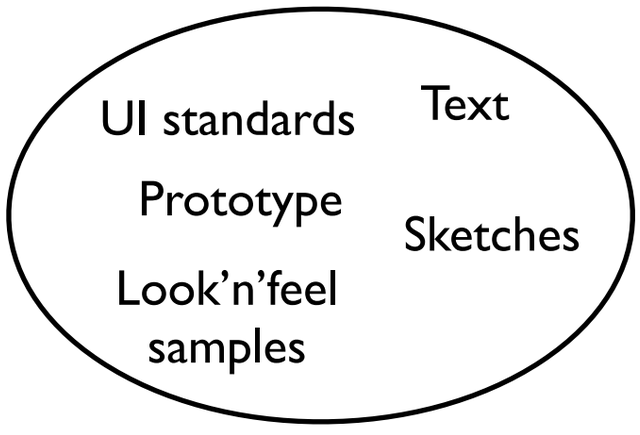
Decision-based



Formal



Quality Requirements



User Interfaces

Prioritization / Triage

- "address only problems/opportunities we have time and resources for"
  - accept that there is no such thing as a _____ solution
  - record _____ between reqs
  - plan more than one _____ ahead
  - plan to _____ before each release
  - goal is to select subset so product can be delivered on ____ and to _____
  - triage participants must see themselves as a ____ and not as part of separate _____
  - both marketing & dev should avoid absolute _____

Change/Management

- "remain flexible as customer and user needs evolve"
  - changes to reqs are ____ not ___
  - do not try to limit the _____ of changes, _____ it
  - meet regularly to decide which reqs are in next _____
  - measure and learn your acceptable change _____ and limit it, or you are likely to fail

NatLangFR

Advantages	
Disadv.	
Efficiency	
Not use	

NatLangFR

Advantages	Flexible, Easy to understand for everyone, Use for any type, Fallback option, Easier use during meetings, No specific knowledge reqs, Easier to version control and prioritize
Disadv.	Ambiguity, Harder to "use" in further dev, Requires language skills, Can lack structure (too flexible), Dependencies harder to track (?), Harder to get overview
Efficiency	Quick, Saves time,
Not use	Some reqs hard in text (UI, QR, Sequences)

Use Cases

Advantages	
Disadv.	
Efficiency	
Not use	

NatLangQR

Advantages	
Disadv.	
Efficiency	
Not use	

PLanguage

Advantages	
Disadv.	
Efficiency	
Not use	

Why is RE hard?

Why is RE hard?

- RE focus on the PROBLEM space rather than SOLUTION space

Why is RE hard?

- RE focus on the PROBLEM space rather than SOLUTION space
  - From “undefined & conflicting ideas” to single/detailed/ technical specification

Why is RE hard?

- RE focus on the PROBLEM space rather than SOLUTION space
  - From “undefined & conflicting ideas” to single/detailed/ technical specification
  - Problem space is much less constrained than solution space

Why is RE hard?

- RE focus on the PROBLEM space rather than SOLUTION space
  - From “undefined & conflicting ideas” to single/detailed/ technical specification
  - Problem space is much less constrained than solution space
  - Reason about System+Environment as a whole and how they affect each other => very complex task

Why is RE hard?

- RE focus on the PROBLEM space rather than SOLUTION space
  - From “undefined & conflicting ideas” to single/detailed/ technical specification
  - Problem space is much less constrained than solution space
  - Reason about System+Environment as a whole and how they affect each other => very complex task
  - Req artefacts must be readable/useable also by non-computing experts => balancing act

Future of RE?

- RE research hot spots [Cheng2007]:
  - Scaling up
  - Tolerance - how to allow “sufficient correctness”
  - Environment understanding and modeling
  - Global SE / Requirements Engineering
  - Methodologies, Patterns and Adapting them
  - Reuse
  - Evaluating effectiveness
  - (Self-managing and self-healing systems)
  - (Security)