

Lecture 10, Part 2: Prioritizing Requirements

Jennifer Campbell
CSC340 - Winter 2007

Why Prioritize Requirements?

- **Planning software development**
 - Scenario: not enough time, budget or resources to implement all requirements listed in an SRS.
 - Requirements in an SRS will be implemented, but the implementation may happen in stages (releases).
 - All requirements are mandatory, but some are essential/critical while others are not.
- **Need to select what to implement**
 - Customers (usually) ask for way too much
 - Balance time-to-market with amount of functionality
 - Decide which features go into the next release

Basics of Prioritization

- **For each requirement/feature, ask:**
 - How important is this to the customer?
 - How much will it cost to implement?
 - How risky will it be to attempt to build it?
- **Perform Triage:**
 - Some requirements
 - **must be included**
 - **should definitely be excluded**
 - Leaving a pool of “**nice-to-haves**”, which we must select from.

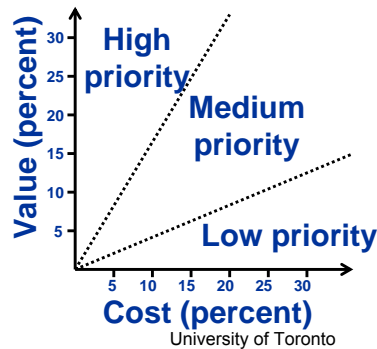


Prioritization Approaches

- **Priority classification**
 - Assign each requirement to a priority classification category
 - Example categories: high, medium, low
- **Cost-Value Approach**
 - Prioritize requirements using ROI (aim for high value, low cost)
- **Others approaches**
 - Quality Function Deployment (QFD)
 - Total Quality Management (TQM)
 - WIN-WIN

Cost-Value Approach

- **Calculate return on investment**
 - Assess each requirement's importance to the project as a whole
 - Assess the relative cost of each requirement
 - Compute the cost-value trade-off:



[KR97]

Estimating Cost & Value

- **Two approaches:**
 - **Absolute scale** (e.g. dollar values)
 - Requires much domain experience
 - **Relative values** (e.g. less/more; a little, somewhat, very)
 - Much easier to elicit
 - Prioritization becomes a sorting problem

Approaches to relative comparisons

- **Basic sort** - for every pair of requirements (i,j), ask if $i > j$?
 - E.g. bubblesort - start in random order, and swap each pair if out of order
 - requires $n*(n-1)/2$ comparisons
- **Construct a Binary Sort Tree**
 - Requires $O(n \log n)$ comparisons
- **Construct a Minimal Spanning Tree**
 - for each pair (R_i, R_{i+1}) get the distance between them
 - Requires $n-1$ comparisons

Prioritization Challenges

- **Hard to quantify differences**
 - easier to say "x is more important than y"...
 - ...than to estimate by how much.
- **Not all requirements comparable**
 - E.g. different level of abstraction
 - E.g. core functionality vs. customer enhancements
- **Requirements may not be independent**
 - No point selecting between X and Y if they are mutually dependent
- **Stakeholders may not be consistent**
 - E.g. If $X > Y$, and $Y > Z$, then presumably $X > Z$?
- **Stakeholders might not agree**
 - Different cost/value assessments for different stakeholders

Cost Value Approach: Analytic Hierarchy Process (AHP)

1. Determine relative value of requirements.

- Stakeholders perform a pair-wise comparison of values.

2. Determine relative cost of requirements.

- Software engineers perform a pair-wise comparison of costs.

3. Prepare a ROI graph.

- Plot the cost (x-axis) and value (y-axis).
- Use the diagram to determine priority (SE and stakeholders).

AHP: 1. Relative value

• Create n x n matrix (for n requirements)

- For element (x,y) in the matrix enter:

- 1 - if x and y are of equal value
- 3 - if x is slightly more preferred than y
- 5 - if x is strongly more preferred than y
- 7 - if x is very strongly more preferred than y
- 9 - if x is extremely more preferred than y
- (use the intermediate values, 2,4,6,8 if compromise needed)

	R1	R2	R3	R4
R1	1	1/3	2	4
R2	3	1	5	3
R3	1/2	1/5	1	1/3
R4	1/4	1/3	3	1

- ...and for (y,x) enter the reciprocal.

[KR97]

AHP: 1. Relative value [2]

• Estimate the Eigenvalues:

- Step 1: Normalize the columns:

- Calculate the sum of each column
- Divide each element in the matrix by the sum of it's column

	R1	R2	R3	R4
R1	1	1/3	2	4
R2	3	1	5	3
R3	1/2	1/5	1	1/3
R4	1/4	1/3	3	1

normalize columns →

	R1	R2	R3	R4
R1	0.21	0.18	0.18	0.48
R2	0.63	0.54	0.45	0.36
R3	0.11	0.11	0.09	0.04
R4	0.05	0.18	0.27	0.12

[KR97]

AHP: 1. Relative value [3]

• Estimate the Eigenvalues:

- Step 2:

- Calculate the sum of each row
- Divide each row sum by the number of rows

	R1	R2	R3	R4
R1	0.21	0.18	0.18	0.48
R2	0.63	0.54	0.45	0.36
R3	0.11	0.11	0.09	0.04
R4	0.05	0.18	0.27	0.12

sum rows and divide by num rows →

sum	sum/4
1.05	0.26
1.98	0.50
0.34	0.09
0.62	0.16

[KR97]

AHP: 1. Relative value [4]

Result of AHP

- the estimated percentage of total value of the project

R1 - 26% of the value
 R2 - 50% of the value
 R3 - 9% of the value
 R4 - 16% of the value

AHP: 1. Relative value [5]

	R1	R2	R3	R4
R1	1	1/3	2	4
R2	3	1	5	3
R3	1/2	1/5	1	1/3
R4	1/4	1/3	3	1

Normalise columns

	R1	R2	R3	R4
R1	0.21	0.18	0.18	0.48
R2	0.63	0.54	0.45	0.36
R3	0.11	0.11	0.09	0.04
R4	0.05	0.18	0.27	0.12

Sum the rows

sum	sum /4
1.05	0.26
1.98	0.50
0.34	0.09
0.62	0.16

Req1 - 26% of the value
 Req2 - 50% of the value
 Req3 - 9% of the value
 Req4 - 16% of the value

Result

AHP: 2. Relative cost

- Follow the same process to determine relative cost as for relative value:

- **Create n x n matrix (for n requirements)**

- For element (x,y) in the matrix enter:
 - 1 - if x and y are of equal cost
 - 3 - if x is slightly more preferred than y
 - 5 - if x is strongly more preferred than y
 - 7 - if x is very strongly more preferred than y
 - 9 - if x is extremely more preferred than y
 - (use the intermediate values, 2,4,6,8 if compromise needed)
- ...and for (y,x) enter the reciprocal.

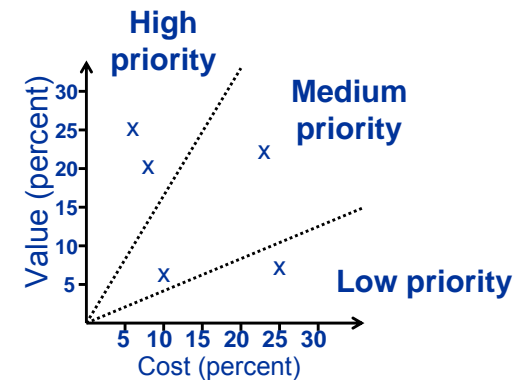
- **Estimate the eigenvalues:**

- E.g. "averaging over normalized columns"
 - Calculate the sum of each column
 - Divide each element in the matrix by the sum of it's column
 - Calculate the sum of each row
 - Divide each row sum by the number of rows

- **Result:** the estimated percentage of total cost of the project

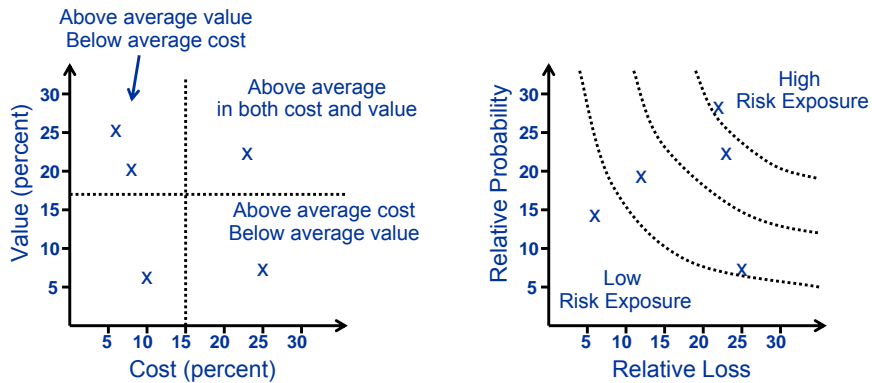
AHP: 3. Plot ROI graph

- ROI Graph:



Other selection criteria

Other ways to group requirements:



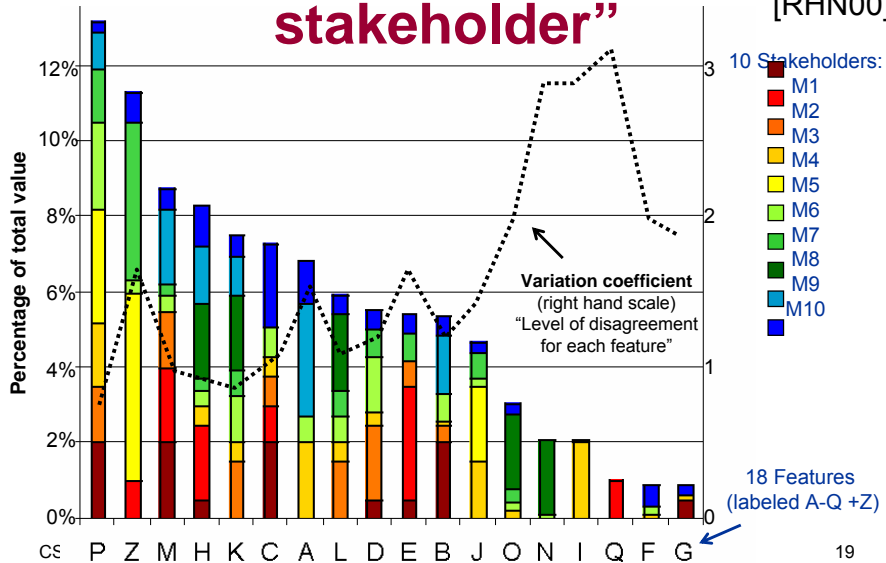
Case Study

- 10 stakeholders
- 17 feature groups
- Each stakeholder prioritized the feature groups
- Ranked the priorities
- Studied how the different stakeholders voted and the resulting priority ranking

[RHN00]

Visualizing "Value by stakeholder"

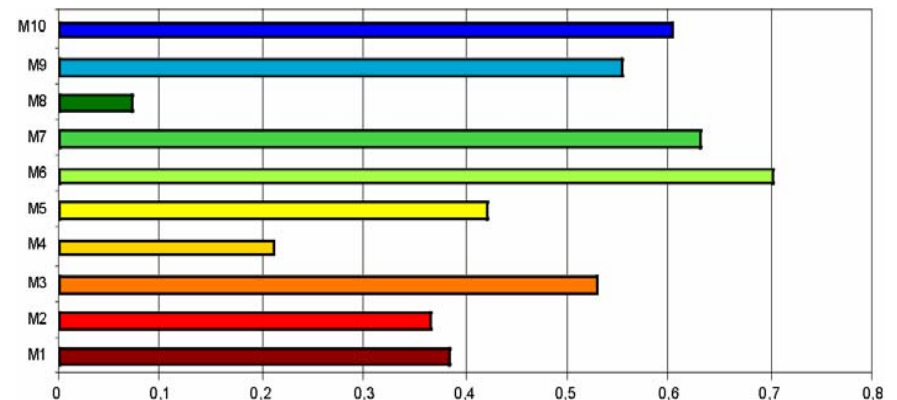
[RHN00]



Visualizing "Stakeholder satisfaction"

[RHN00]

- Graph showing correlation between stakeholder's priorities and the group's priorities
 - Can also be thought of as "influence of each stakeholder on the group"



Weighting Stakeholders

- To prioritize requirements accurately, may need to value stakeholders differently
 - E.g. to reflect credibility?
 - E.g. to reflect size of constituency represented?

- **Weighting Stakeholders Priorities**
 - Assign a weight to each stakeholder.
 - The sum of the weights is equal to 1.



$$p_i = \sum_{k=1}^m w_k \times p_{ik}$$

- p_i is the priority of requirement i
- w_k is the weight of stakeholder k
- p_{ik} is the priority of stakeholder k on requirement i
- m is the number of stakeholders

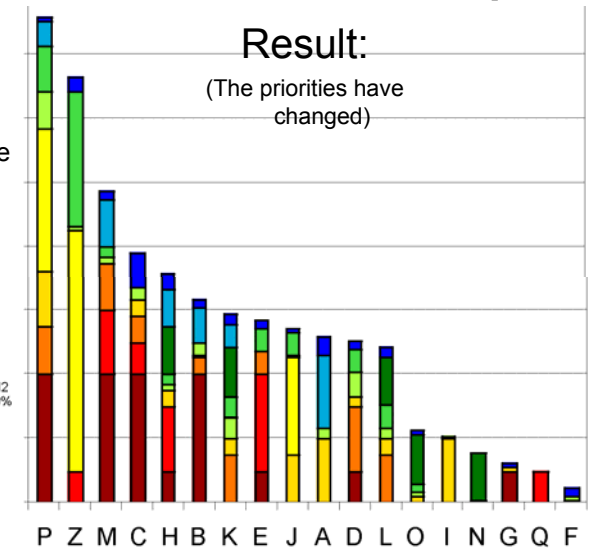
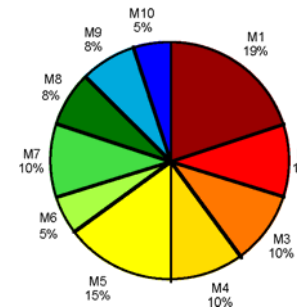
[RHN00]

21

Weighting Stakeholders

[RHN00]

- Weight each stakeholder
 - E.g. to reflect credibility?
 - E.g. to reflect size of constituency represented?



Basic approaches to conflict resolution

- **Defining Conflict**
 - In RE, focus typically is on logical inconsistency:
 - E.g. conflict is a divergence between goals - there is a feasible boundary condition that makes the goals inconsistent [van Lamsweerde et al. 1998]
 - Note:
 - conflict may occur between individuals, groups, organizations, or different roles played by one person
- **Resolution Method:**
 - Three broad types of resolution method can be distinguished:
 - **Co-operative (or collaborative) methods**, which include negotiation and education;
 - **Competitive methods**, which include combat, coercion and competition;
 - **Third Party methods**, which include arbitration and appeals to authority.

- **Negotiation**
 - ...is collaborative exploration:
 - participants attempt to find a settlement that satisfies all parties as much as possible.
 - also known as:
 - integrative behaviour
 - constructive negotiation
 - distinct from:
 - distributive/competitive negotiation
- **Competition**
 - is maximizing your own gain:
 - no regard for the degree of satisfaction of other parties.
 - but not necessarily hostile!
 - Extreme form:
 - when all gains by one party are at the expense of others
- **Third Party Resolution**
 - participants appeal to outside source
 - the rule-book, a figure of authority, or the toss of a coin.
 - can occur with the breakdown of either negotiation or competition as resolution methods.
 - types of third party resolution
 - judicial: cases presented by each participant are taken into account
 - extra-judicial: a decision is determined by factors other than the cases presented (e.g. relative status of participants).
 - arbitrary: e.g. toss of a coin

References

- [KR97] Karlsson, J. and Ryan, K. 1997. *A Cost-Value Approach for Prioritizing Requirements*. IEEE Software. 14, 5 (Sep. 1997), pages 67-74.
- [RHN00] Regnell, B., Host, M, Natt och Dag, J., Beremark, P., and Thomas, H. *Visualization of Agreement and Satisfaction in Distributed Prioritization of Market Requirements*. REFSQ 2000.