Slicing systems – work breakdown structure

Slice vertical; not horizontal. This means that rather than slicing systems into subsystems, or infrastructure pieces (e.g. complete data base design) we want to slice the system into small user visible functional pieces, so that each piece goes from UI through the business logic, to the database and back.

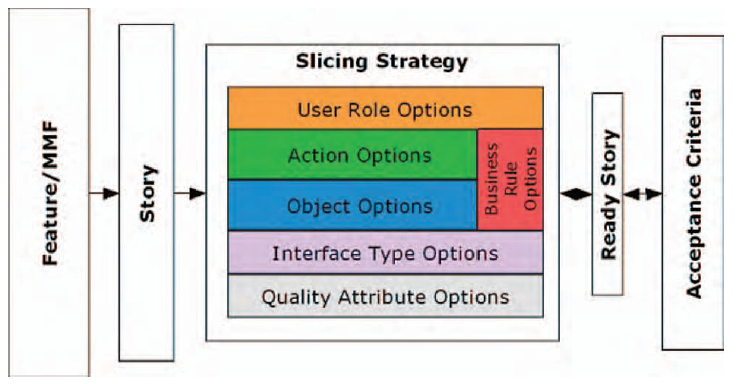
Requirements fundamentals

1. Iteratively/Incrementally developed – start with a vision and list of use cases. Use cases are refined and sliced as needed.
2. Finished in order. Limit work in progress. Deliver fast and often.
3. Progress is shown to the stakeholders frequently.
4. Grouped into releases according to a product roadmap. Goal is continuous deployment.
5. Ordered so as to maximize long term return on investment (ROI)
6. Negotiable
7. Items yet to be developed are continuously reordered, reworked, and updated.
8. Items ready to be developed are vertically sliced into stories/scenarios that can be finished within 2 team days.
9. Only developed when the ROI is positive
10. Non- functional requirements are specified and continuously tested

Slicing requirements

I like to structure functional requirements as use cases

1. Use the complete template and for each scenario do not neglect the risk criteria, preconditions, post conditions, and fundamental interactions.
2. Use variations of the pre-conditions to generate alternate scenarios
3. If the base scenario is still too large, slice it using the techniques below:



Action Items could include CRUD (Create, Read, Update, and Delete) actions