

# Analysis Roadmap

The following table contains an idealised breakdown of the activities, techniques and deliverables that come from the Requirements Gathering and Analysis Workflows. Idealised, because it is very difficult to capture the iterative and incremental nature of the process using what is essentially a linear representation (i.e. a table), especially when the number and scale of iterations cannot be predicted. So bear in mind that some of the activities may occur concurrently rather than sequentially, and revisited when appropriate and necessary.

Activity	Technique	Identifies	Deliverable
<b>Inception Phase</b>			
Define Problem		Sponsor Nature of problem Constraints on Solution	Problem Statement Vision Statement Glossary*
Scoping	Some form of facilitated 'Brainstorming' session involving all stakeholders. cf Joint Application Development (JAD) sessions	Business Processes Actors Required system behaviours System boundary Interfaces	Business Context Diagram Activity Diagrams (Business Processes) Use Case Diagram
Specify Requirements - High Level	As above	Key behavioural descriptions Errors/Exceptions Pre/postconditions (system states) Inclusions/Extensions/Generalisations	Use Case Descriptions with Activity Diagrams (Use Case flows) System state model (if required)
<b>Elaboration Phase</b>			
Identify and define key domain concepts/candidate objects	Inspection, domain interviews		Extended Glossary* Conceptual Schema
Classify objects	Inspection, semantic analysis, domain knowledge, common sense	(Some) classes, attributes, operations, roles	Class Diagram (showing classes only) Repository*
Specify Requirements - Detailed	Role Play using Class, Responsibility, Collaborator (CRC) cards (not a RUP/UML technique, but useful for discovering classes, collaborations, behaviours. Generates scenario subject-verb-object 'scripts')	Classes, their responsibilities (attributes/operations) and collaborators (relationships) New/redundant classes New errors/exceptions Sub-scenarios	At least $1 + n$ Scenarios for each Use Case, expressed in terms of key objects Refined Class Diagram* Refined Repository* Refined Use Cases*
Construct graphical representations of Scenarios	Sequence Diagrams Communication Diagrams		Sequence Diagrams Communication Diagrams Refined Scenarios*

Optimise and refine Scenarios and Sequence Diagrams	Parameterisation, message/response pairing	Reusable requirement 'chunks' (sub-scenarios) Operation signatures Attributes Functional dependencies	Refined Sequence/Communication Diagrams* Refined Class Diagram (associations, attributes and operations)
Optimise and refine Relationships	Name relationships, identify roles, ask 'how many?', test boundary conditions, identify construction/destruction dependencies	Multiplicity, semantics, constraints, reflexive associations, aggregations, attributed associations	Scenarios to test constraints Refined Class Diagram (aggregations, multiplicity, roles)
Generalise	Identify common attributes, behaviours, relationships, semantics. Consider contribution to overall reuse architecture	Inheritance hierarchies	Refined Class Diagram (inheritance)
Test and present Model	Trace each scenario through class diagram, present model to peers, challenge everything	Faults, problems, shortcomings, improvements	Narrative Refined Class Diagram Complete, correct and consistent set of Use Cases Testing Strategy/Test Cases