Considerations when Building an Expert system

- We will describe the necessary prerequisites when building an expert system.

- The system should be a quality product.

- The development should be cost effective and timely.

- Designing of expert systems of part of a general field known as Knowledge Management (KM).

Selecting the Appropriate Problem

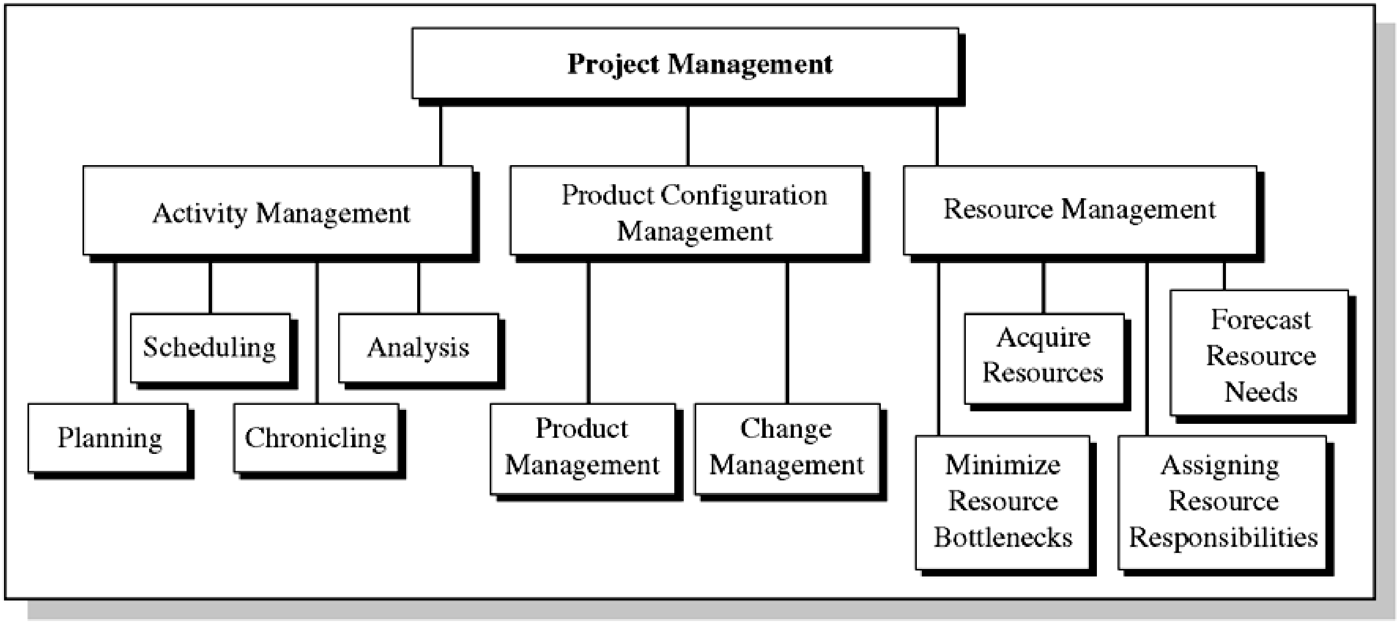
- We need to answer the questions, “Why are we building this expert system?”.
  - Intellectual Property Agreement must be considered
  - Clearly identify the problem
  - Clearly identify the expert
  - Clearly identify the users

- We need to know the payoff – money, efficiency, etc.

Selecting the Appropriate Problem

- What tools will be available to build the expert system?
  - Check the Web for applications in existence
  - Know the language necessary to create a semantic net of relationships on which the system will be based
- How much will the expert system cost?
  - A function of people, resources, time, etc.
  - How available is the knowledge?

Figure 6.1 Project Management Tasks



Stages in the Development

- How will the system be developed?
  - This will depend on the resources provided

Stages:

1. Feasibility Study – see if the project is feasible
2. Rapid Prototype – demonstrate ideas / impress
3. Refined System – verification by knowledge engineers
4. Field Testable – system tested by selected users

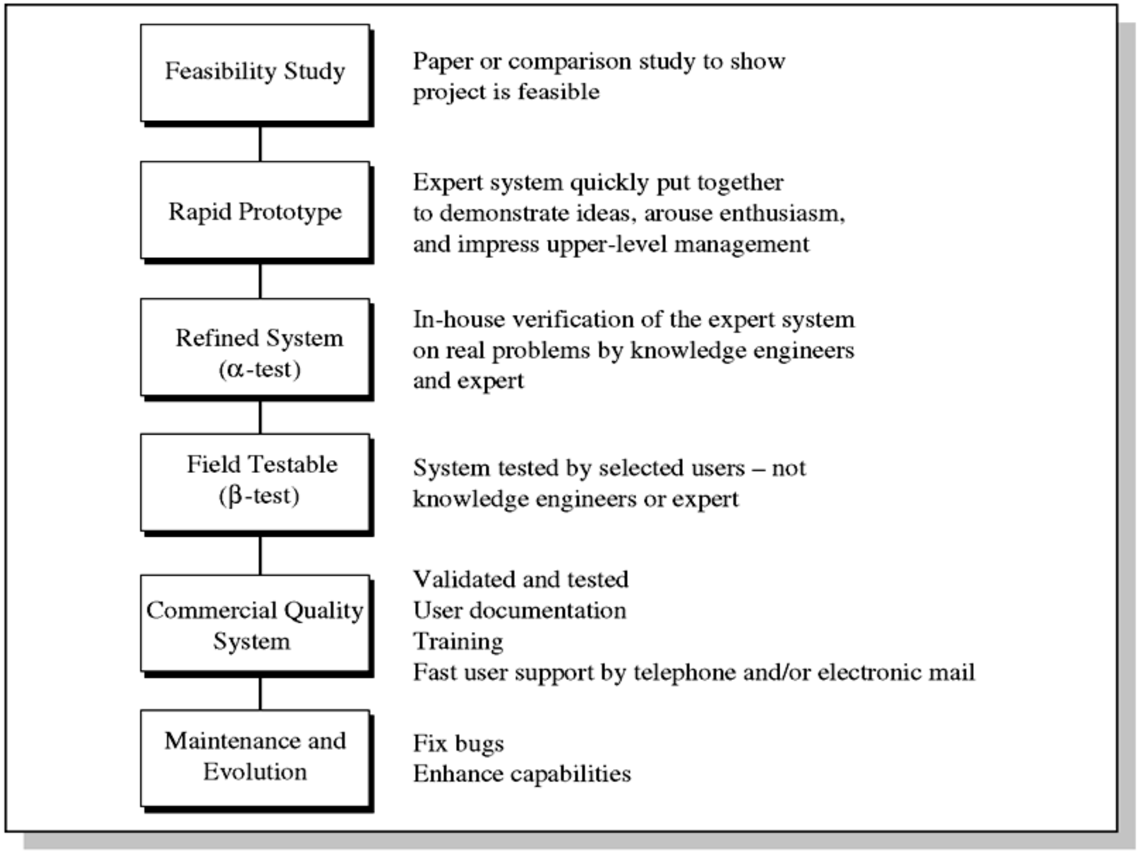
Stages in the Development

5. Commercial quality system – validation / testing
6. Maintenance and evolution – repair bugs, enhance capabilities

Other Considerations

- How will the system be delivered?
  - Should be considered in earliest stages of development
  - Integration with existing programs
- How will the system be maintained and evolve?
  - Performance is dependent on knowledge / expertise
  - Performance must be maintained
  - New knowledge will be acquired
  - Old knowledge will be modified

Figure 6.2 General Stages in the Development of an Expert System



Feasibility Study

Paper or comparison study to show project is feasible

Rapid Prototype

Expert system quickly put together to demonstrate ideas, arouse enthusiasm, and impress upper-level management

Refined System (α-test)

In-house verification of the expert system on real problems by knowledge engineers and expert

Field Testable (β-test)

System tested by selected users – not knowledge engineers or expert

Commercial Quality System

Validated and tested
User documentation
Training
Fast user support by telephone and/or electronic mail

Maintenance and Evolution

Fix bugs
Enhance capabilities

Errors in Development Stages

- Expert’s knowledge may be erroneous, propagating errors throughout the entire development process.
  - Formal procedures may be necessary to certify expert
  - Technique panels can scrutinize expert’s knowledge
  - Focus groups can also be used

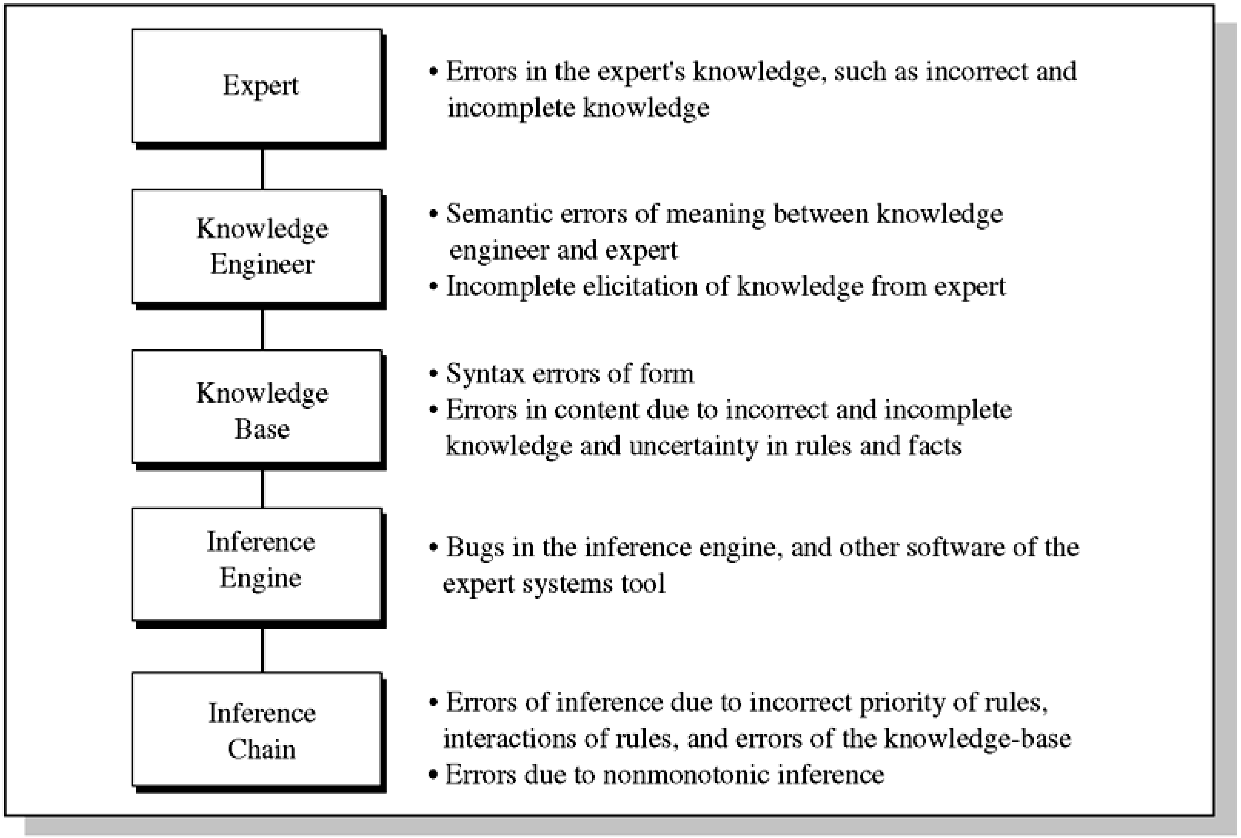
Errors in the Development Stages

- Knowledge may not be properly communicated to knowledge engineer, or knowledge may be misinterpreted.
- Knowledge base may be corrupted by entering incorrect form of a rule or fact.

Errors in the Development Stages

- Inference engine errors may result from errors in pattern matching, conflict resolution, and execution of actions.
- Inference chain errors may be caused by erroneous knowledge, semantic errors, inference engine bugs, incorrect specifications of rule priorities, and strange interaction among rules.
- Limits of ignorance – a problem common to all previous stages

Figure 6.3 Major Errors in Expert Systems and Some Causes



Expert

- Errors in the expert's knowledge, such as incorrect and incomplete knowledge

Knowledge Engineer

- Semantic errors of meaning between knowledge engineer and expert
- Incomplete elicitation of knowledge from expert

Knowledge Base

- Syntax errors of form
- Errors in content due to incorrect and incomplete knowledge and uncertainty in rules and facts

Inference Engine

- Bugs in the inference engine, and other software of the expert systems tool

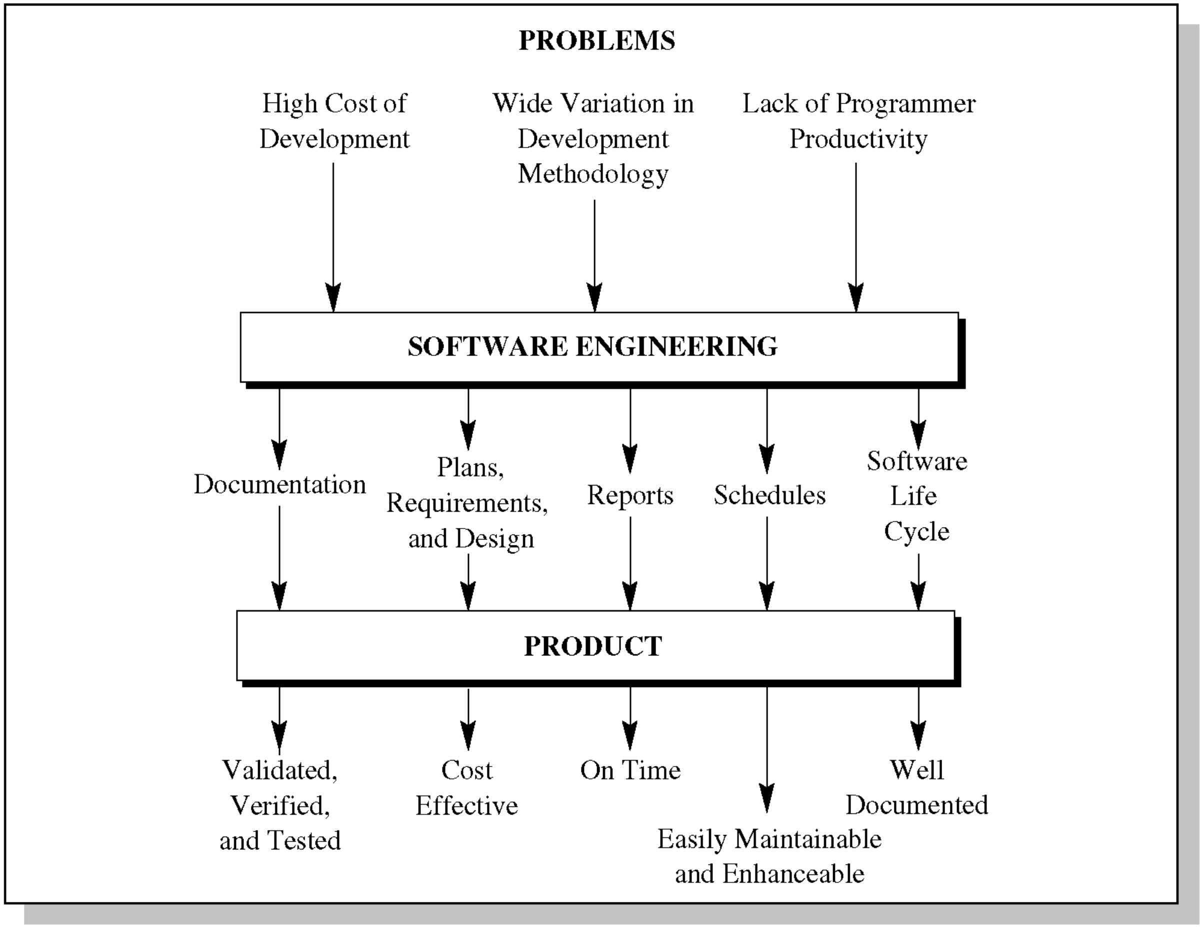
Inference Chain

- Errors of inference due to incorrect priority of rules, interactions of rules, and errors of the knowledge-base
- Errors due to nonmonotonic inference

Software Engineering and Expert Systems

- Expert systems are products like any other software product and require good standards for development.
- Expert systems may have serious responsibilities – life and death.
- High standards are a necessity and can be measured by “mean time between failures”.

Figure 6.4 Software Engineering Methodology


PROBLEMS
High Cost of Development
Wide Variation in Development Methodology
Lack of Programmer Productivity
SOFTWARE ENGINEERING
Documentation
Plans, Requirements, and Design
Reports
Schedules
Software Life Cycle
PRODUCT
Validated, Verified, and Tested
Cost Effective
On Time
Easily Maintainable and Enhanceable
Well Documented

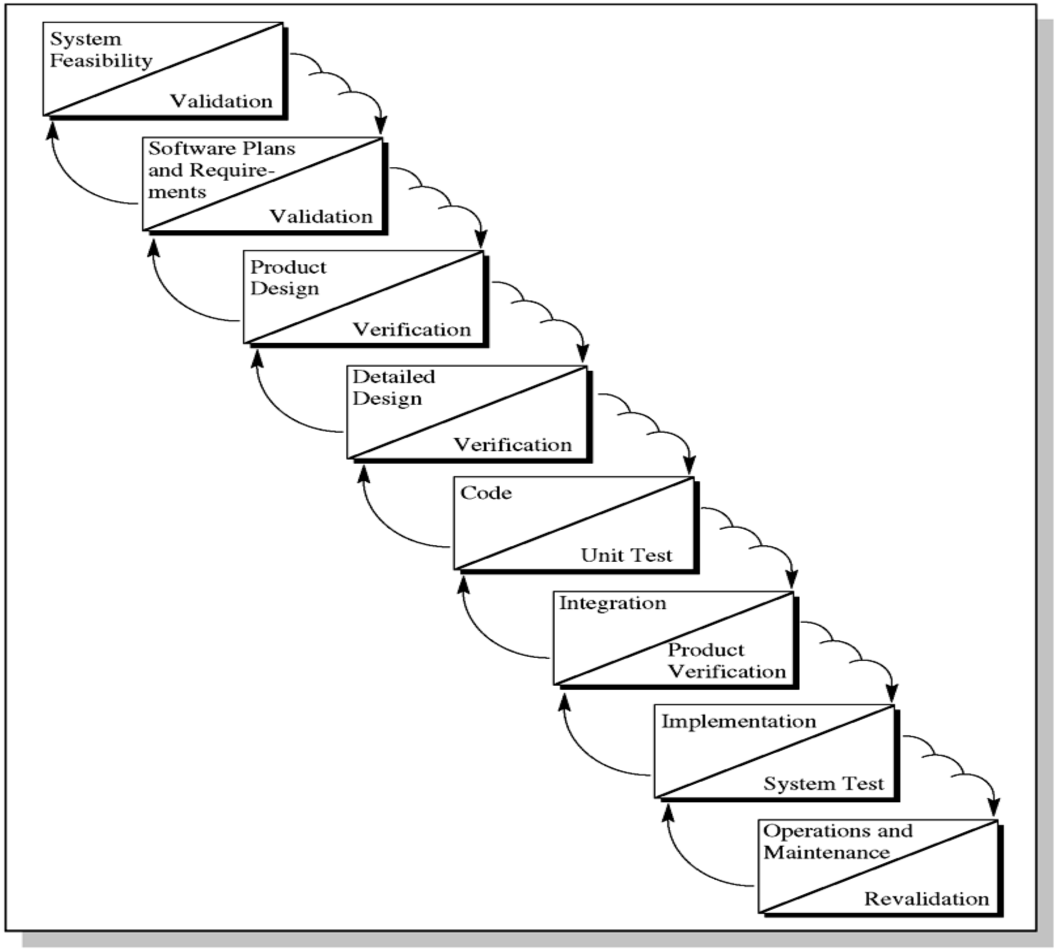
Expert System Life Cycle

- Begins with the initial concept of the software and ends with its retirement from use.
- Expert systems require more maintenance because they are based on knowledge that is:
  - Heuristic
  - Experiential
- A number of life cycle models have been developed.

Waterfall Model

- Each stage ends with a verification and validation activity to minimize any problems in that stage.
- Arrows go back and forth only one stage at a time.
- It is assumed that all information necessary for a stage is known.

Figure 6.5 Waterfall Model of the Software Life Cycle



Code-and-Fix Model

- Some code is written and then fixed when it does not work correctly.
- Usually the method of choice for new programming students in conventional and expert systems
- This eventually led to the do-it-twice concept where a prototype then a final system was built.

Incremental Model

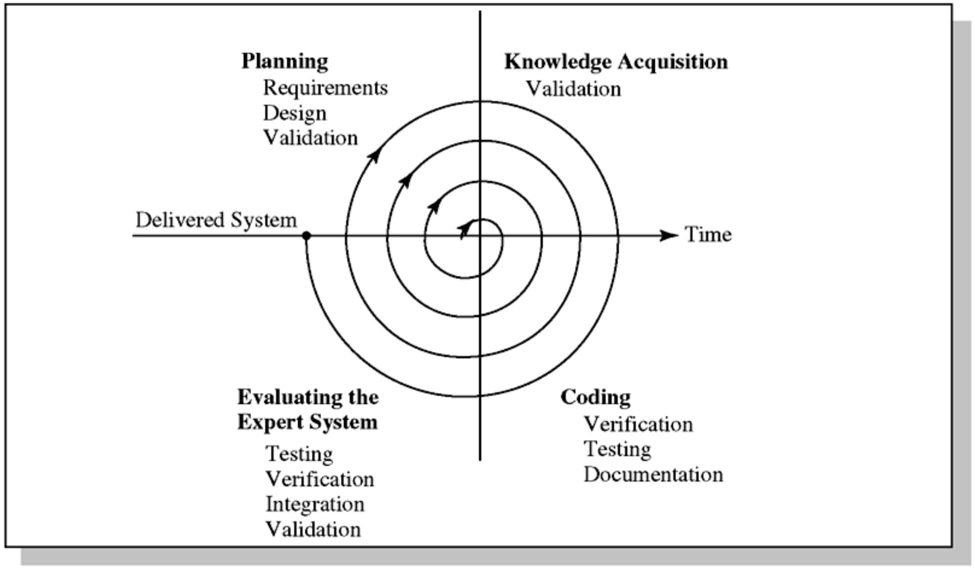
- This is a refinement of the waterfall and top-down-approach.

- The idea is to develop software in increments of functional capability.
  - Major increment – assistant → colleague → expert
  - Minor increment – expertise w/in each level
  - Microincrement – add/refining individual rules

Spiral Model

Each circuit of the spiral adds some functional capability to the system.

Figure 6.6 A Spiral Model of Expert System Development

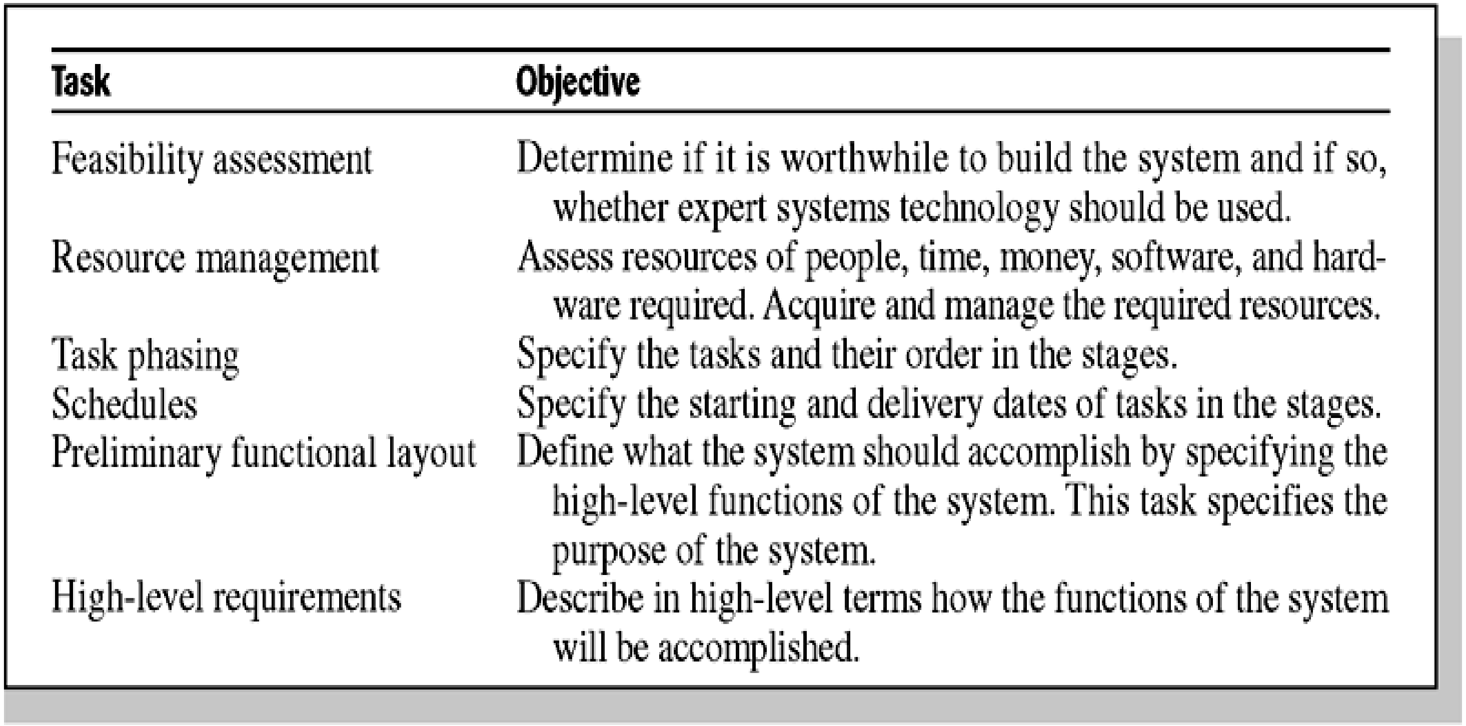


Detailed Life Cycle Model
Spiral Model

1. Planning Stage

The purpose of this stage is to produce a formal work plan for the expert system development – documents to guide and evaluate the development.

Table 6.2 Planning Stage Tasks



Task	Objective
Feasibility assessment	Determine if it is worthwhile to build the system and if so, whether expert systems technology should be used.
Resource management	Assess resources of people, time, money, software, and hardware required. Acquire and manage the required resources.
Task phasing	Specify the tasks and their order in the stages.
Schedules	Specify the starting and delivery dates of tasks in the stages.
Preliminary functional layout	Define what the system should accomplish by specifying the high-level functions of the system. This task specifies the purpose of the system.
High-level requirements	Describe in high-level terms how the functions of the system will be accomplished.

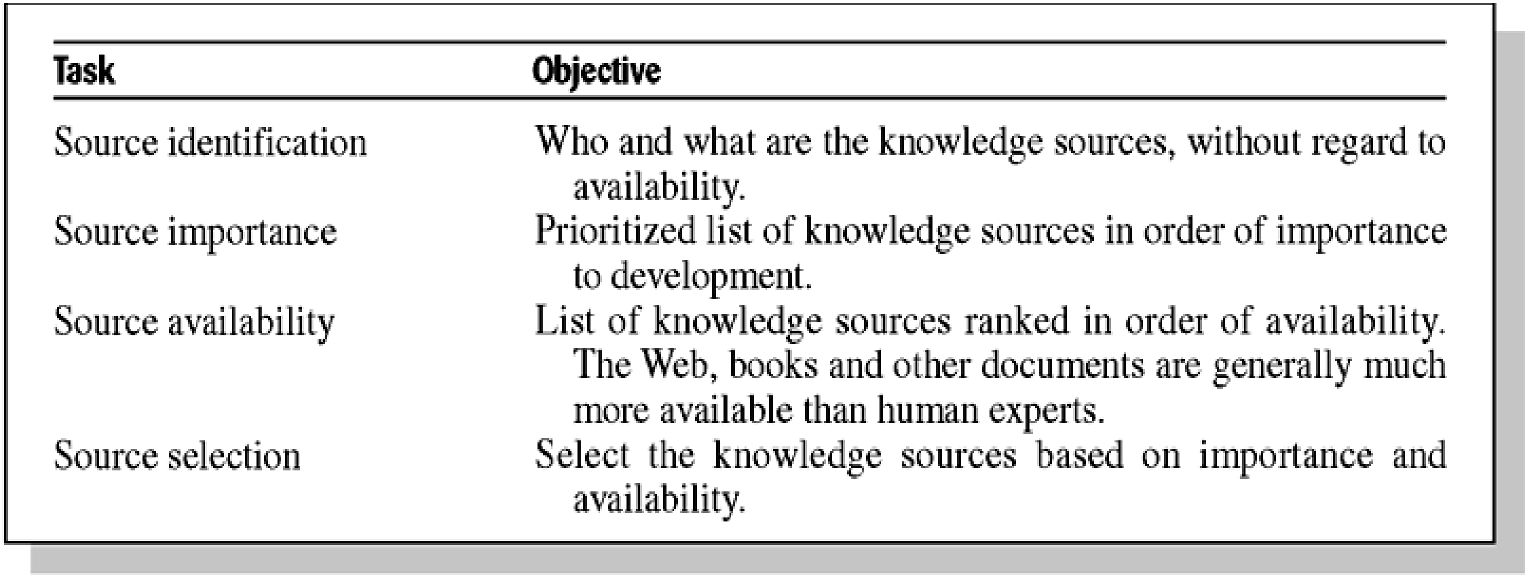
Linear Model

2. Knowledge Definition

The objective of this stage is to define the knowledge requirements of the expert system, which consists of two main tasks:

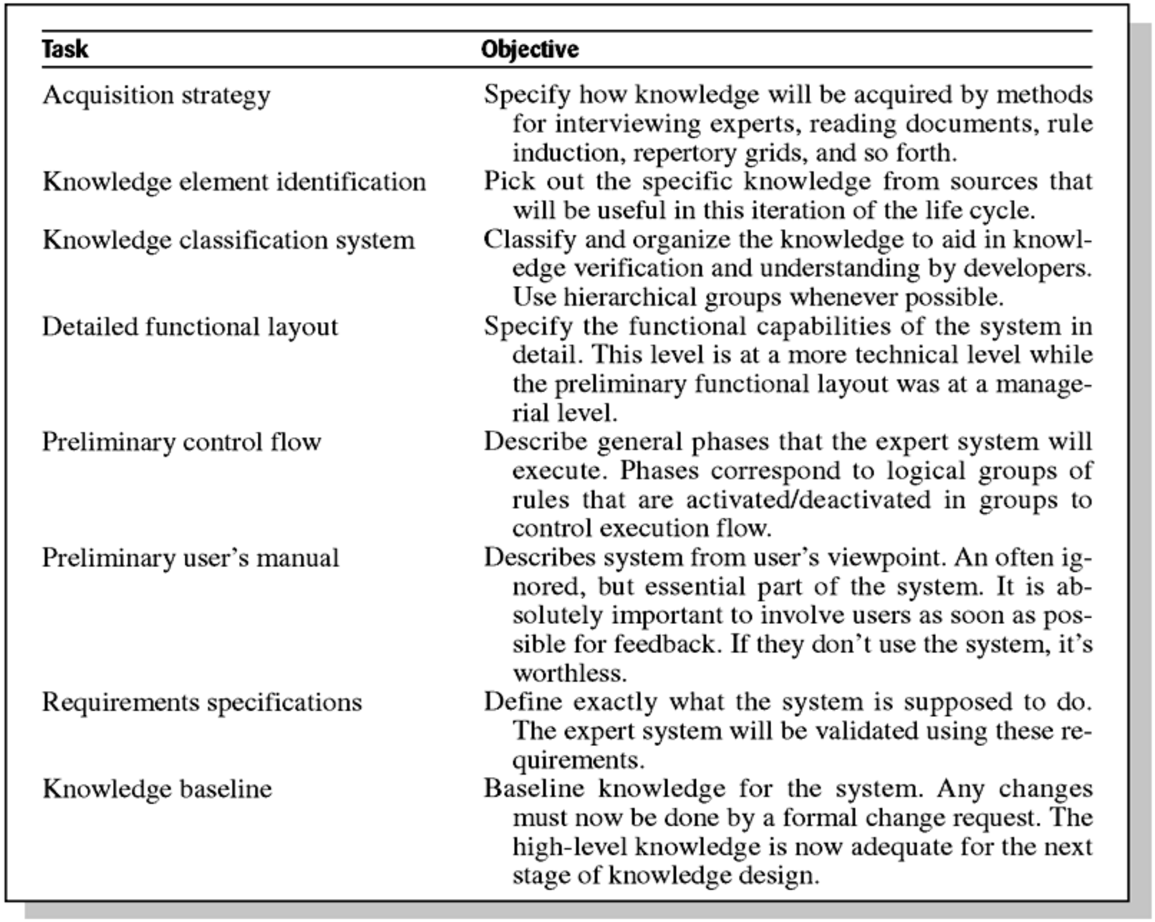
- Knowledge source identification and selection
- Knowledge acquisition, analysis, and extraction

Table 6.3 Knowledge Source / Identification



Task	Objective
Source identification	Who and what are the knowledge sources, without regard to availability.
Source importance	Prioritized list of knowledge sources in order of importance to development.
Source availability	List of knowledge sources ranked in order of availability. The Web, books and other documents are generally much more available than human experts.
Source selection	Select the knowledge sources based on importance and availability.

Table 6.4 Knowledge Acquisition, Analysis, and Extraction Tasks



Task	Objective
Acquisition strategy	Specify how knowledge will be acquired by methods for interviewing experts, reading documents, rule induction, repertory grids, and so forth.
Knowledge element identification	Pick out the specific knowledge from sources that will be useful in this iteration of the life cycle.
Knowledge classification system	Classify and organize the knowledge to aid in knowledge verification and understanding by developers. Use hierarchical groups whenever possible.
Detailed functional layout	Specify the functional capabilities of the system in detail. This level is at a more technical level while the preliminary functional layout was at a managerial level.
Preliminary control flow	Describe general phases that the expert system will execute. Phases correspond to logical groups of rules that are activated/deactivated in groups to control execution flow.
Preliminary user's manual	Describes system from user's viewpoint. An often ignored, but essential part of the system. It is absolutely important to involve users as soon as possible for feedback. If they don't use the system, it's worthless.
Requirements specifications	Define exactly what the system is supposed to do. The expert system will be validated using these requirements.
Knowledge baseline	Baseline knowledge for the system. Any changes must now be done by a formal change request. The high-level knowledge is now adequate for the next stage of knowledge design.

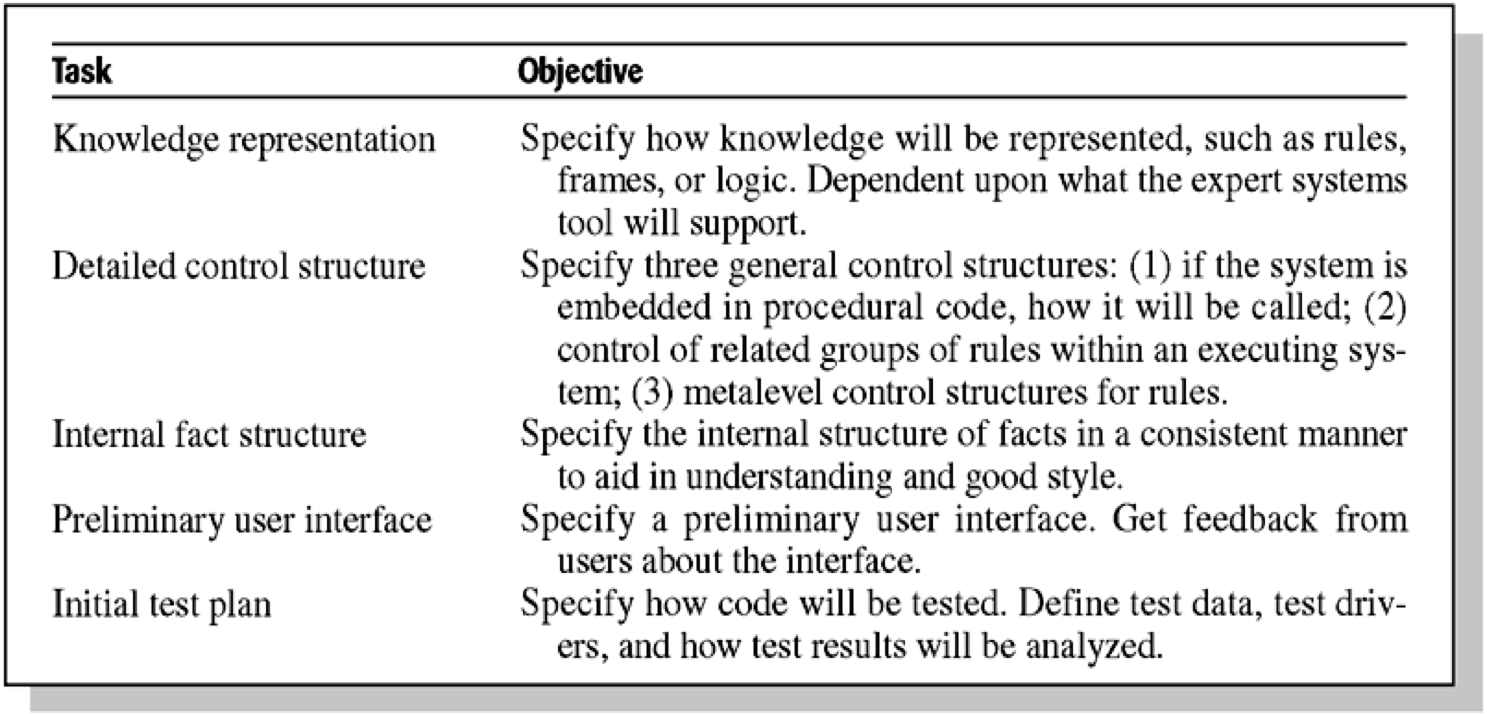
Spiral Model

3. Knowledge Design

The objective is to produce the detailed design for an expert system and involves:

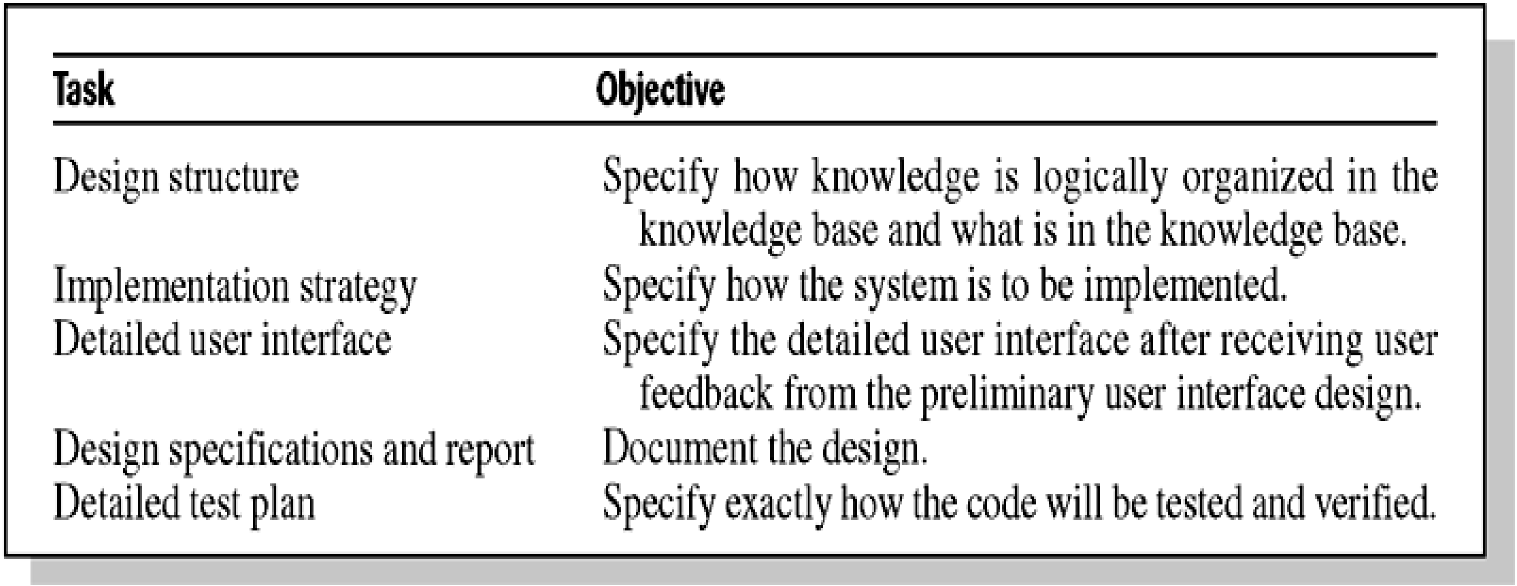
- Knowledge definition
- Detailed design

Table 6.5 Knowledge Definition Tasks



Task	Objective
Knowledge representation	Specify how knowledge will be represented, such as rules, frames, or logic. Dependent upon what the expert systems tool will support.
Detailed control structure	Specify three general control structures: (1) if the system is embedded in procedural code, how it will be called; (2) control of related groups of rules within an executing system; (3) metalevel control structures for rules.
Internal fact structure	Specify the internal structure of facts in a consistent manner to aid in understanding and good style.
Preliminary user interface	Specify a preliminary user interface. Get feedback from users about the interface.
Initial test plan	Specify how code will be tested. Define test data, test drivers, and how test results will be analyzed.

Table 6.6 Detailed Design of Knowledge Tasks



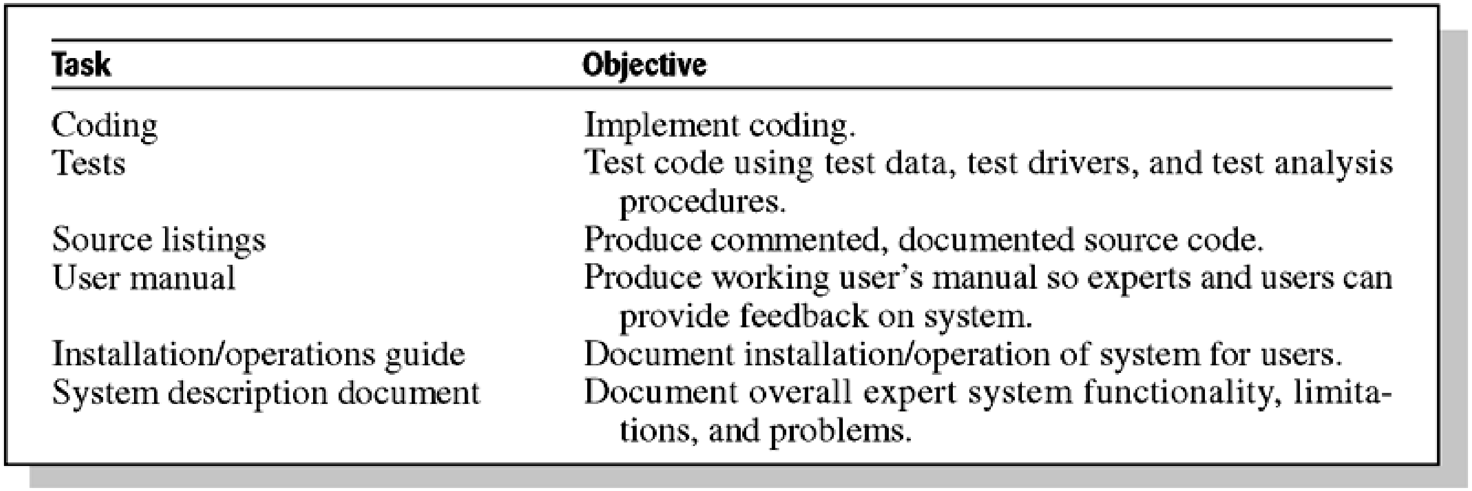
Task	Objective
Design structure	Specify how knowledge is logically organized in the knowledge base and what is in the knowledge base.
Implementation strategy	Specify how the system is to be implemented.
Detailed user interface	Specify the detailed user interface after receiving user feedback from the preliminary user interface design.
Design specifications and report	Document the design.
Detailed test plan	Specify exactly how the code will be tested and verified.

Linear Model

4. Code and Checkout

This begins the actual code implementation

Table 6.7 Code and Checkout Tasks



Task	Objective
Coding	Implement coding.
Tests	Test code using test data, test drivers, and test analysis procedures.
Source listings	Produce commented, documented source code.
User manual	Produce working user's manual so experts and users can provide feedback on system.
Installation/operations guide	Document installation/operation of system for users.
System description document	Document overall expert system functionality, limitations, and problems.

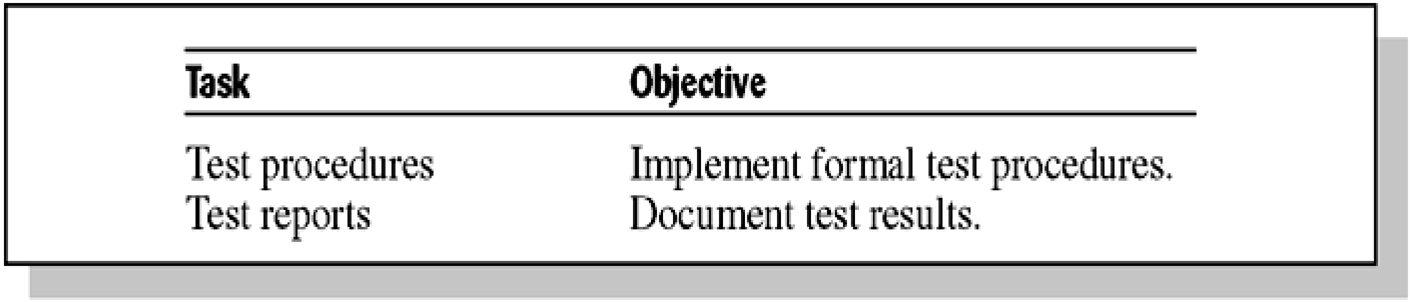
Linear Model

5. Knowledge Verification

The objective here is to determine the correctness, completeness, and consistency of the system.

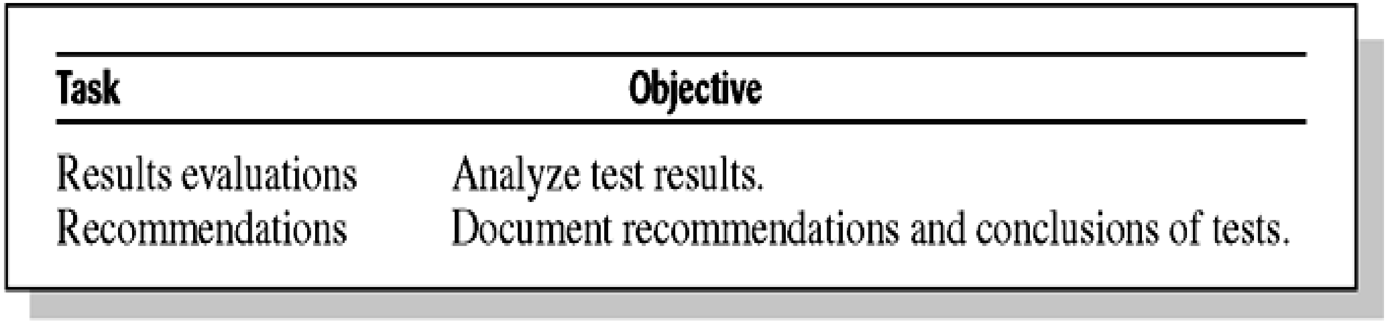
- Formal tests
- Test Analysis

Table 6.8 Formal Test Tasks of Knowledge Verification Stage



Task	Objective
Test procedures	Implement formal test procedures.
Test reports	Document test results.

Test Analysis Tasks



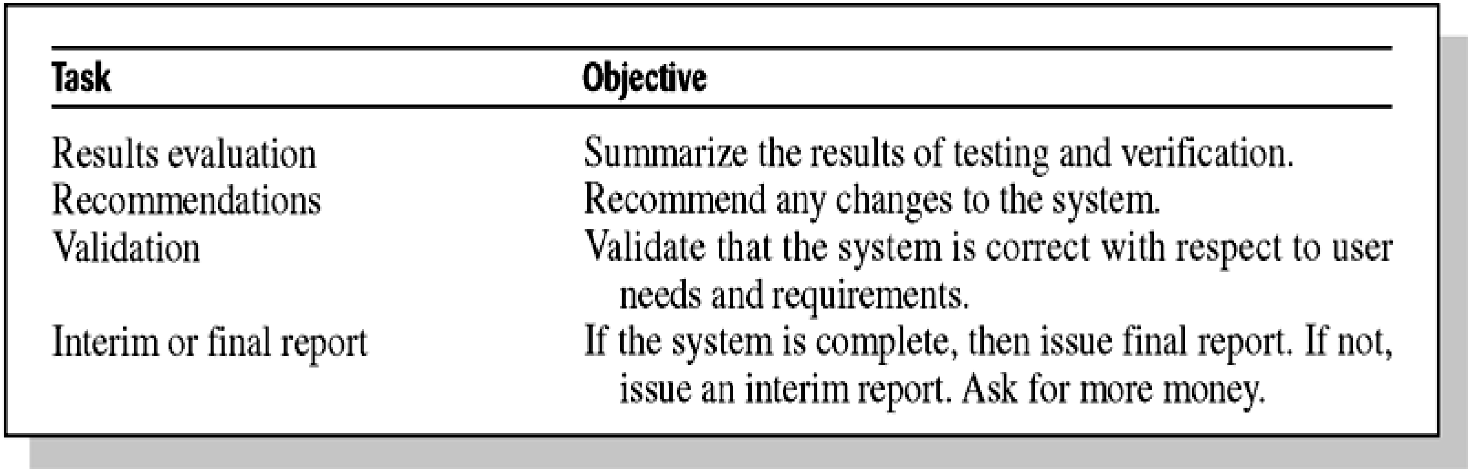
Task	Objective
Results evaluations	Analyze test results.
Recommendations	Document recommendations and conclusions of tests.

Linear Model

6. System Evaluation

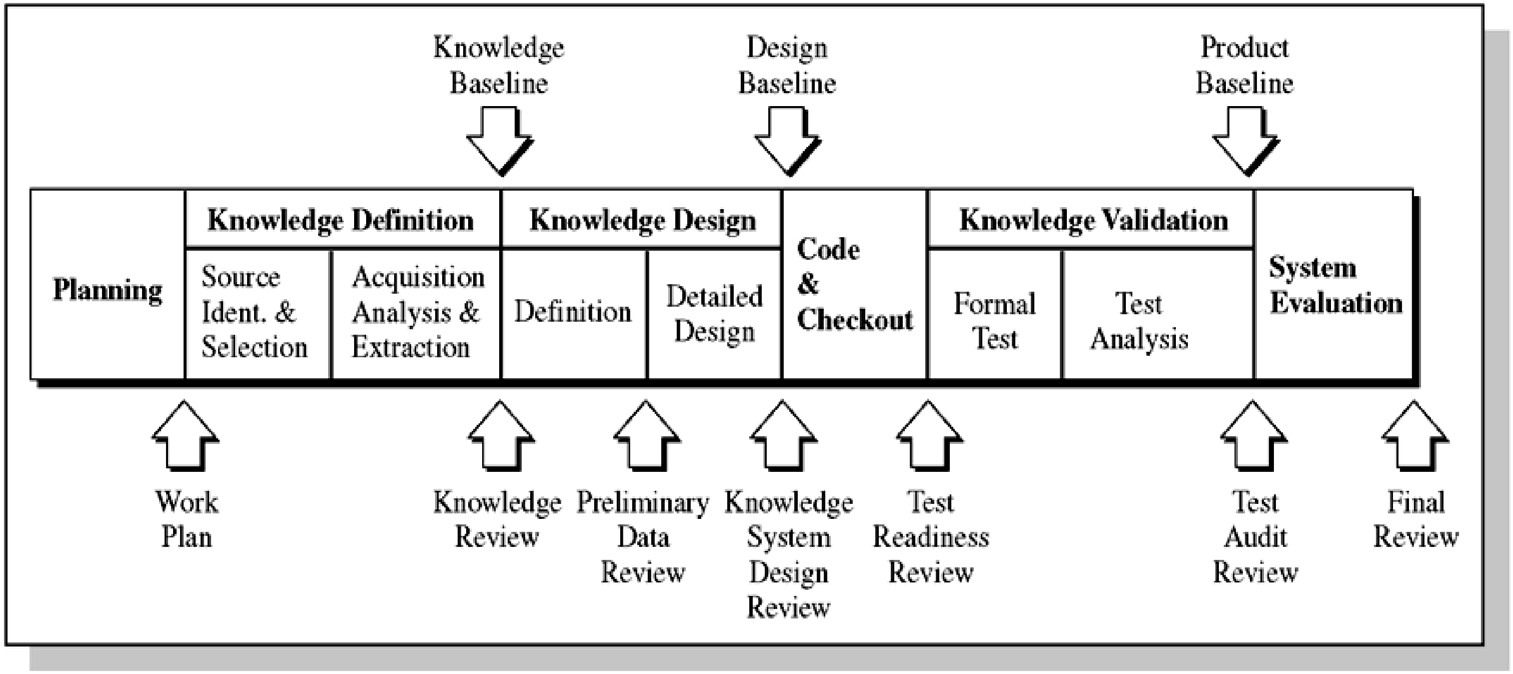
   This stage is for summarizing what has been learned with recommendations for improvements and corrections.

Table 6.10 System Evaluation Stage Tasks



Task	Objective
Results evaluation	Summarize the results of testing and verification.
Recommendations	Recommend any changes to the system.
Validation	Validate that the system is correct with respect to user needs and requirements.
Interim or final report	If the system is complete, then issue final report. If not, issue an interim report. Ask for more money.

Figure 6.7 Linear Model of Expert System Development Life Cycle



Knowledge Baseline
Design Baseline
Product Baseline
Planning
Knowledge Definition
Source Ident. & Selection
Acquisition Analysis & Extraction
Knowledge Design
Definition
Detailed Design
Code & Checkout
Knowledge Validation
Formal Test
Test Analysis
System Evaluation
Work Plan
Knowledge Review
Preliminary Data Review
Knowledge System Design Review
Test Readiness Review
Test Audit Review
Final Review