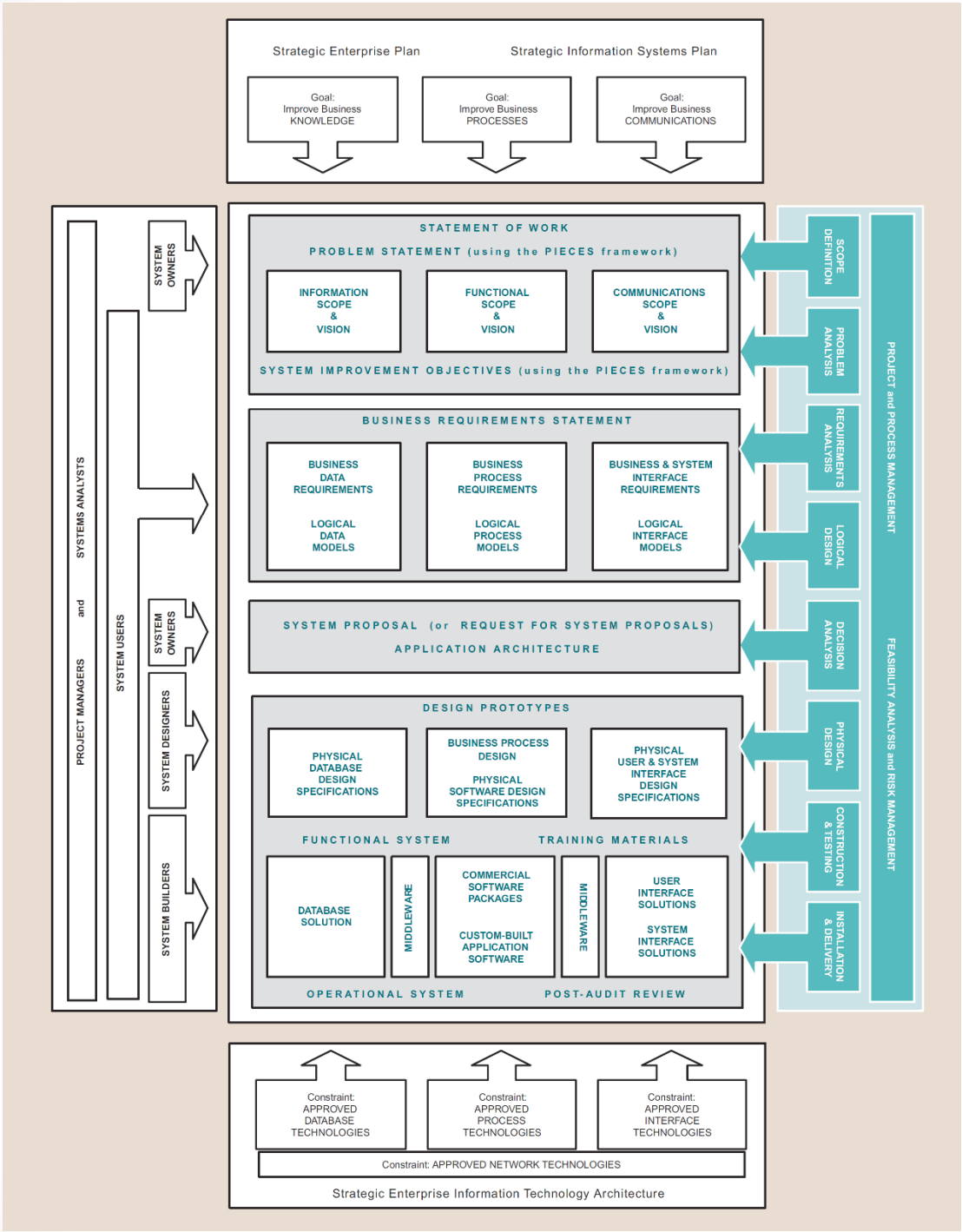
System Analysis and Design

Developing Information Systems

By: Vahid Rahimian

Spring 2022


Strategic Enterprise Plan
Strategic Information Systems Plan
Goal: Improve Business KNOWLEDGE
Goal: Improve Business PROCESSES
Goal: Improve Business COMMUNICATIONS
PROJECT MANAGERS and SYSTEMS ANALYSTS
SYSTEM USERS
SYSTEM OWNERS
SYSTEM OWNERS
SYSTEM DESIGNERS
SYSTEM BUILDERS
STATEMENT OF WORK
PROBLEM STATEMENT (using the PIECES framework)
INFORMATION SCOPE & VISION
FUNCTIONAL SCOPE & VISION
COMMUNICATIONS SCOPE & VISION
SYSTEM IMPROVEMENT OBJECTIVES (using the PIECES framework)
BUSINESS REQUIREMENTS STATEMENT
BUSINESS DATA REQUIREMENTS
LOGICAL DATA MODELS
BUSINESS PROCESS REQUIREMENTS
LOGICAL PROCESS MODELS
BUSINESS & SYSTEM INTERFACE REQUIREMENTS
LOGICAL INTERFACE MODELS
SYSTEM PROPOSAL (or REQUEST FOR SYSTEM PROPOSALS)
APPLICATION ARCHITECTURE
DESIGN PROTOTYPES
PHYSICAL DATABASE DESIGN SPECIFICATIONS
BUSINESS PROCESS DESIGN
PHYSICAL SOFTWARE DESIGN SPECIFICATIONS
PHYSICAL USER & SYSTEM INTERFACE DESIGN SPECIFICATIONS
FUNCTIONAL SYSTEM
TRAINING MATERIALS
DATABASE SOLUTION
MIDDLEWARE
COMMERCIAL SOFTWARE PACKAGES
CUSTOM-BUILT APPLICATION SOFTWARE
MIDDLEWARE
USER INTERFACE SOLUTIONS
SYSTEM INTERFACE SOLUTIONS
OPERATIONAL SYSTEM
POST-AUDIT REVIEW
SCOPE DEFINITION
PROBLEM ANALYSIS
REQUIREMENTS ANALYSIS
LOGICAL DESIGN
DECISION ANALYSIS
PHYSICAL DESIGN
CONSTRUCTION & TESTING
INSTALLATION & DELIVERY
PROJECT and PROCESS MANAGEMENT
FEASIBILITY ANALYSIS and RISK MANAGEMENT
Constraint: APPROVED DATABASE TECHNOLOGIES
Constraint: APPROVED PROCESS TECHNOLOGIES
Constraint: APPROVED INTERFACE TECHNOLOGIES
Constraint: APPROVED NETWORK TECHNOLOGIES
Strategic Enterprise Information Technology Architecture

Process of System Development

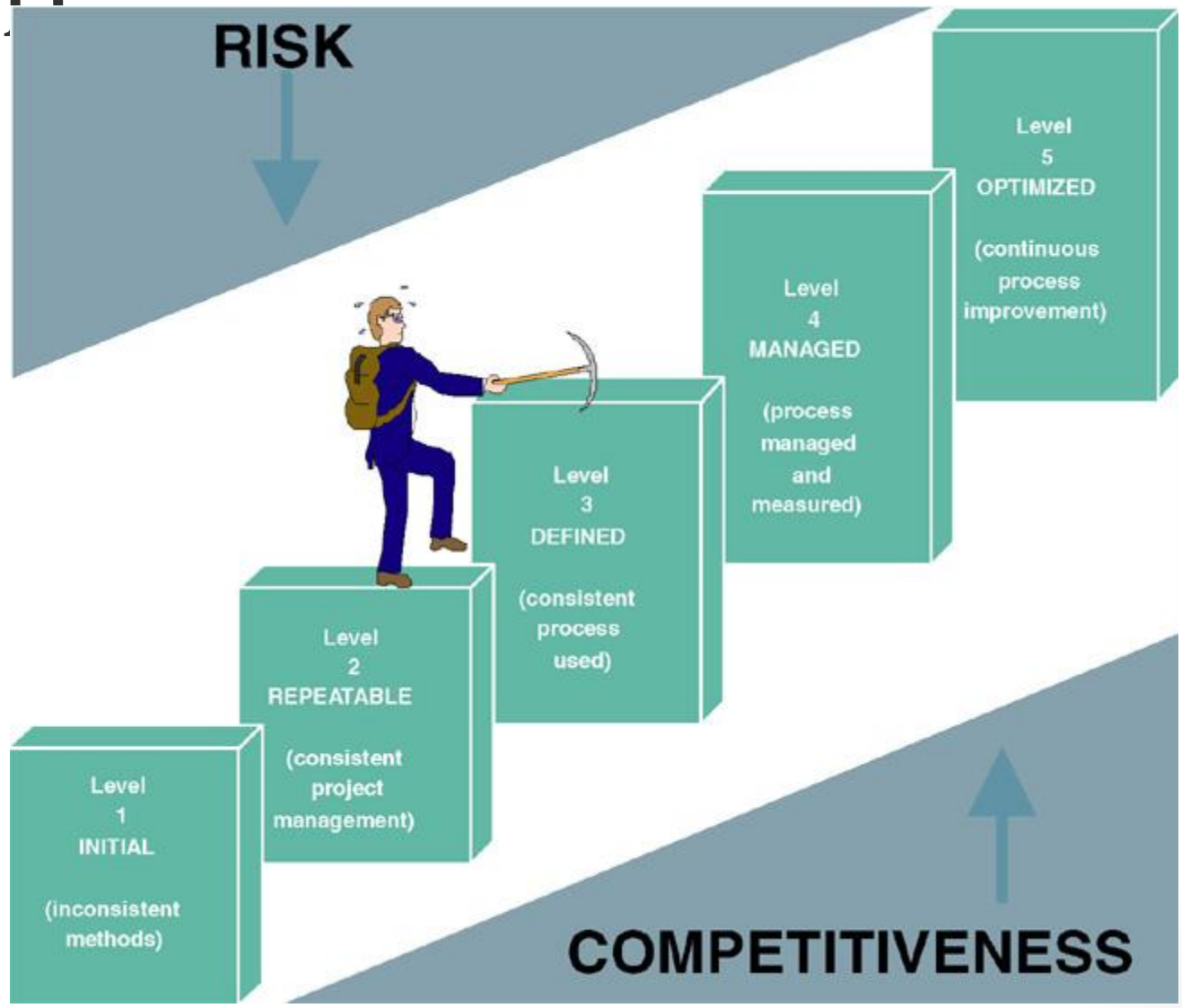
- **System development process** – a set of activities, methods, best practices, deliverables, and automated tools that stakeholders use to develop and continuously improve information systems and software.
  - Many variations
  - Using a consistent process for system development:
    - Create efficiencies that allow management to shift resources between projects
    - Produces consistent documentation that reduces lifetime costs to maintain the systems
    - Promotes quality

CMM Process Management Model

Capability Maturity Model (CMM) – a standardized framework for assessing the maturity level of an organization's information system development and management processes and products. It consists of five levels of maturity:

- **Level 1—Initial**: System development projects follow no prescribed process.
- **Level 2—Repeatable**: Project management processes and practices established to track project costs, schedules, and functionality.
- **Level 3—Defined**: Standard system development process (methodology) is purchased or developed. All projects use a version of this process.
- **Level 4—Managed**: Measurable goals for quality and productivity are established.
- **Level 5—Optimizing**: The standardized system development process is continuously monitored and improved based on measures and data analysis established in Level 4.

Capability Maturity Model (CMM)



Impact of System Development "Process" on Quality

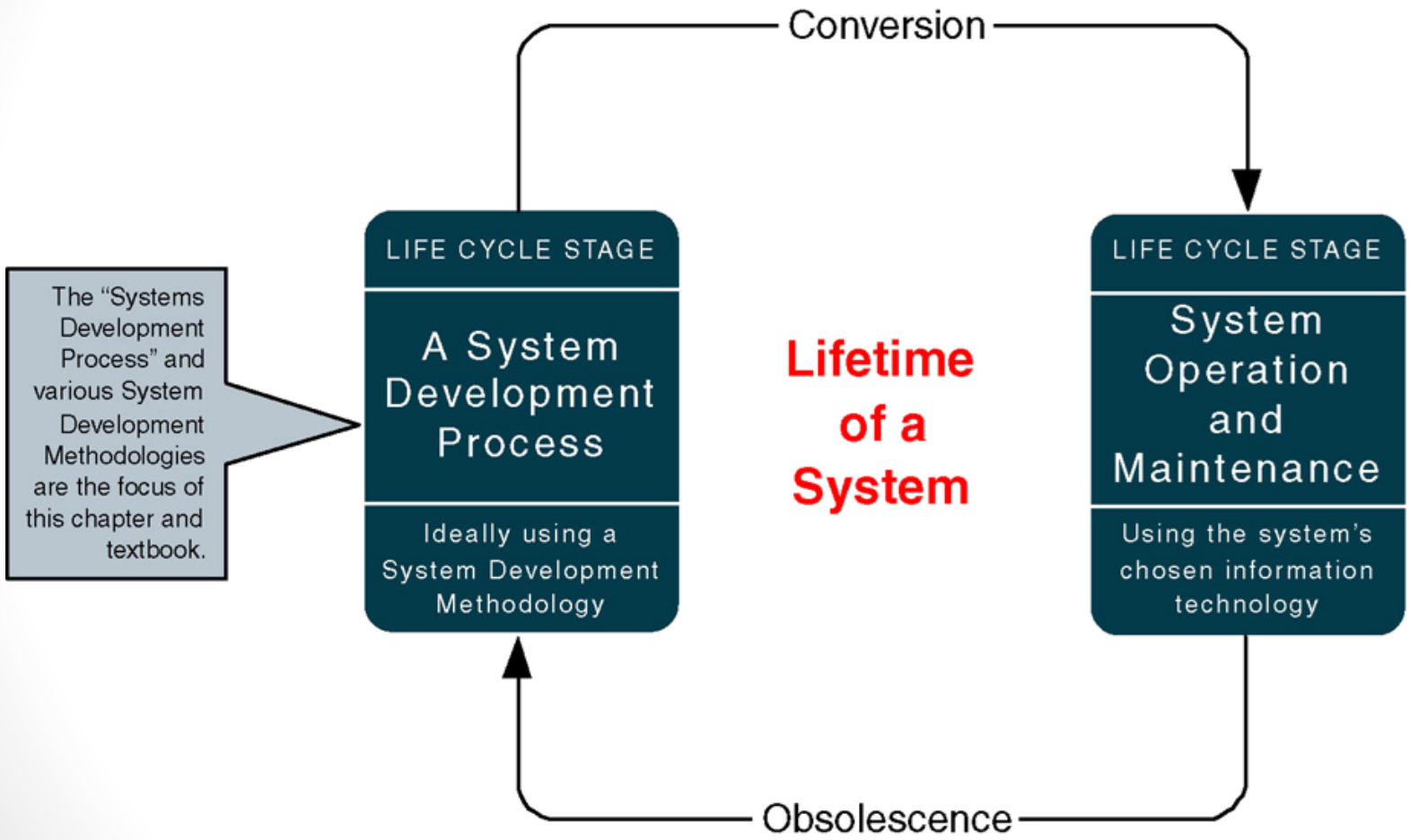
CMM Project Statistics for a Project Resulting in 200,000 Lines of Code						
Organization's CMM Level	Project Duration (months)	Project Person-Months	Number of Defects Shipped	Median Cost ($ millions)	Lowest Cost ($ millions)	Highest Cost ($ millions)
1	30	600	61	5.5	1.8	100+
2	18.5	143	12	1.3	.96	1.7
3	15	80	7	.728	.518	.933

Life Cycle versus Methodology

- **System life cycle** – the factoring of the lifetime of an information system into two stages, (1) systems development and (2) systems operation and maintenance.

- **System development methodology** – a formalized approach to the systems development process; a standardized development process that defines (as in CMM Level 3) a set of activities, methods, best practices, deliverables, and automated tools that system developers and project managers are to use to develop and continuously improve information systems and software.

A System Life Cycle



Representative System Development Methodologies

- Architected Rapid Application Development (Architected RAD)
- Dynamic Systems Development Methodology (DSDM)
- Joint Application Development (JAD)
- Information Engineering (IE)
- Rapid Application Development (RAD)
- Rational Unified Process (RUP)
- Structured Analysis and Design
- eXtreme Programming (XP)

Representative System Development Methodologies

- Feature Driven Development (FDD)
- Lean Software Development
- Scrum
- Kanban
- Scaled Agile Framework (SAFe)
- Large-scale Scrum (LeSS)

- There are ‘practices’ like CI/CD, TDD, DDD
- There are ‘patterns’ like CQRS, Publisher-Subscriber, Sharding, …

Principles of System Development

- Get the system users involved.
- Use a problem-solving approach.
- Establish phases and activities.
- Document through development.
- Establish standards.
- Manage the process and projects
- Justify systems as capital investments.
- Don’t be afraid to cancel or revise scope.
- Divide and conquer.
- Design systems for growth and change.

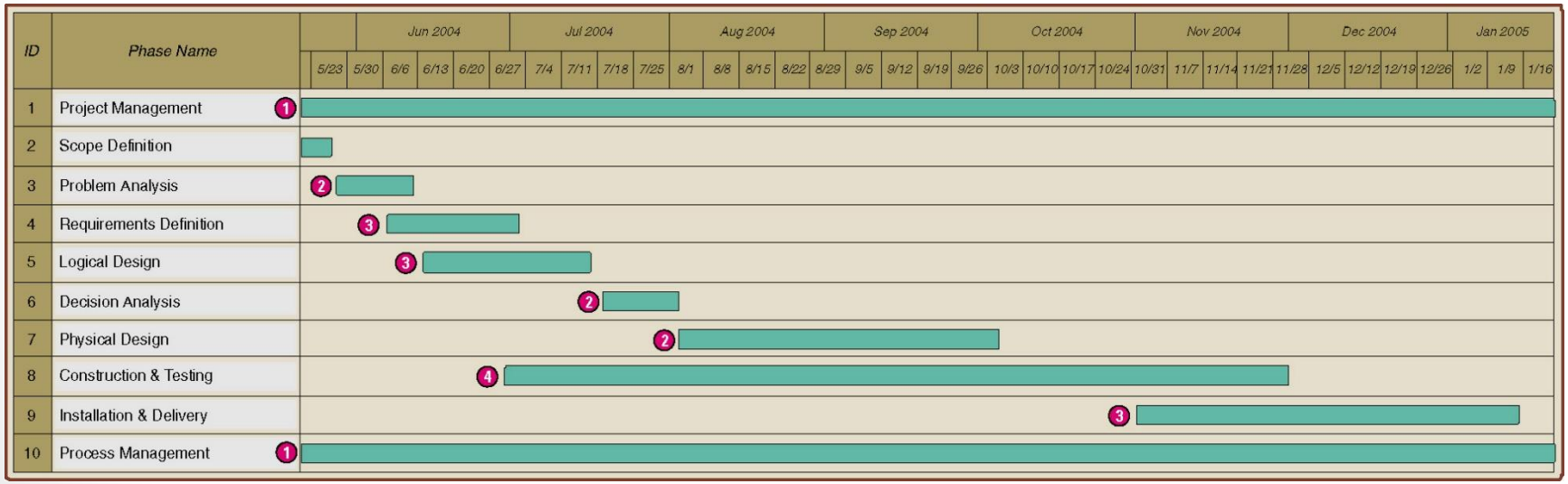
Use a Problem-Solving Approach

Classical Problem-solving approach

1. Study and understand the problem, its context, and its impact.
2. Define the requirements that must be meet by any solution.
3. Identify candidate solutions that fulfill the requirements, and select the “best” solution.
4. Design and/or implement the chosen solution.
5. Observe and evaluate the solution’s impact, and refine the solution accordingly.

Establish Phases and Activities

- **Overlap of System Development Phases**



Manage the Process and Projects

- **Process management** – an ongoing activity that documents, manages, oversees the use of, and improves an organization's chosen methodology (the "process") for system development. Process management is concerned with phases, activities, deliverables, and quality standards should be consistently applied to all projects.

- **Project management** is the process of scoping, planning, staffing, organizing, directing, and controlling a project to develop an information system at a minimum cost, within a specified time frame, and with acceptable quality.

Justify Information Systems as Capital Investments

- **Cost-effectiveness** – The result obtained by striking a balance between the lifetime costs of developing, maintaining, and operating an information system and the benefits derived from that system. Cost-effectiveness is measured by a cost-benefit analysis.

- **Strategic information systems plan** – a formal strategic plan (3-5 years) for building and improving an information technology infrastructure and the information system applications that use that infrastructure.

- **Strategic enterprise plan** – a formal strategic plan (3-5 years) for an entire business that defines its mission, vision, goals, strategies, benchmarks, and measures of progress and achievement. Usually, the strategic enterprise plan is complemented by strategic business unit plans that define how each business unit will contribute to the enterprise plan. The information systems plan is one of those unit-level plans.

Don’t Be Afraid to Cancel or Revise Scope

- **Creeping commitment** – a strategy in which feasibility and risks are continuously reevaluated throughout a project. Project budgets and deadlines are adjusted accordingly.

- **Risk management** – the process of identifying, evaluating, and controlling what might go wrong in a project before it becomes a threat to the successful completion of the project or implementation of the information system. Risk management is driven by risk analysis or assessment.

Where Do Systems Development Projects Come From?

- Planned Projects
  - An **information systems strategy plan** has examined the business as a whole to identify those system development projects that will return the greatest strategic (long-term) value to the business
  - A **business process redesign** has thoroughly analyzed a series of business processes to eliminate redundancy and bureaucracy and to improve efficiency and value added. Not it is time to redesign the supporting information system for those redesigned business processes.

Where Do Systems Development Projects Come From?

- Unplanned projects
  - Triggered by a specific problem, opportunity, or directive that occurs in the course of doing business.
  - **Steering committee** – an administrative body of system owners and information technology executives that prioritizes and approves candidate system development projects.
  - **Backlog** – a repository of project proposals that cannot be funded or staffed because they are a lower priority than those that have been approved for system development.

The PIECES Problem-Solving Framework

Problem, Opportunity & Directive identification

P	the need to improve performance
I	the need to improve information (and data)
E	the need to improve economics, control costs, or increase profits
C	the need to improve control or security
E	the need to improve efficiency of people and processes
S	the need to improve service to customers, suppliers, partners, employees, etc.

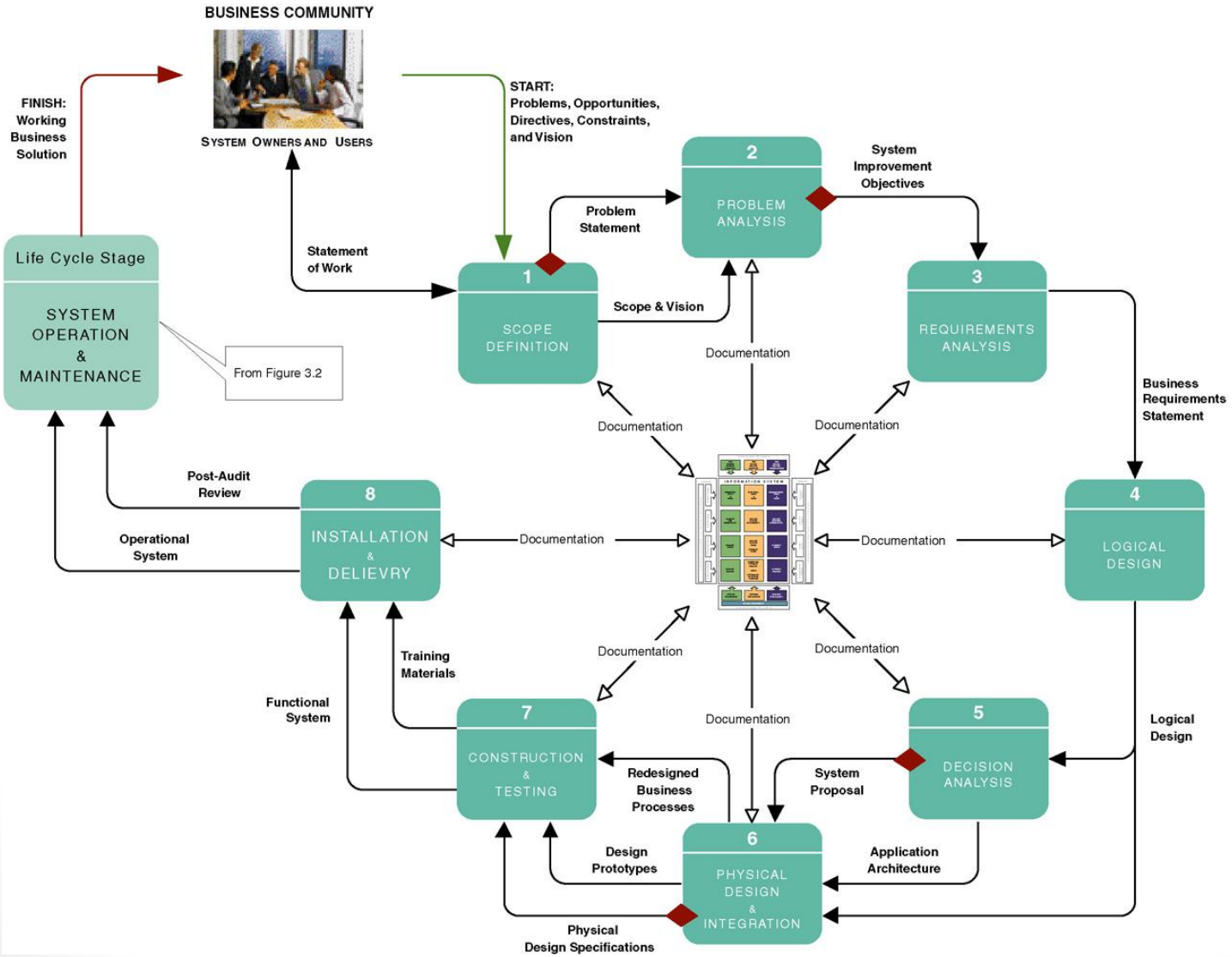
Project Phases

- **FAST** - (Framework for the Application of Systems Thinking) a hypothetical methodology used throughout this book to demonstrate a representative systems development process.
- Each methodology will use different project phases.

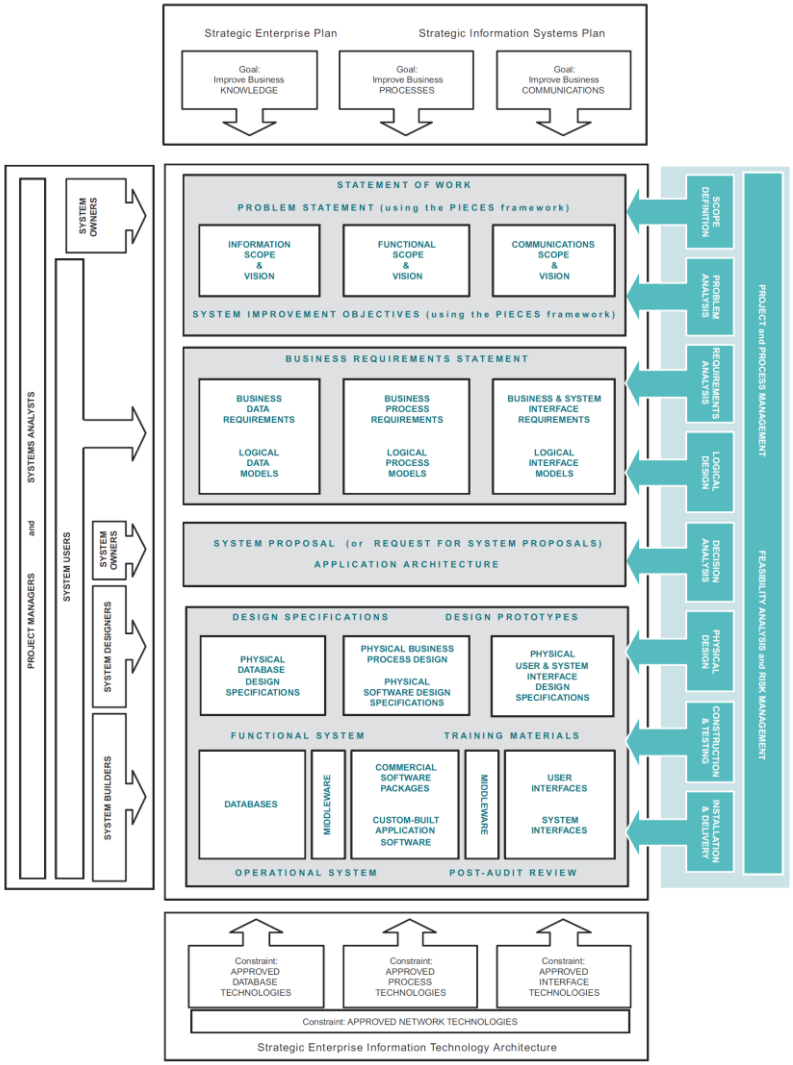
Project Phases

FAST Phases	Classic Phases (from Chapter 1)			
	Project Initiation	System Analysis	System Design	System Implementation
Scope Definition	X			
Problem Analysis	X	X		
Requirements Analysis		X		
Logical Design		X		
Decision Analysis	(a system analysis transition phase)			
Physical Design and Integration			X	
Construction and Testing			X	X
Installation and Delivery				X

FAST Project Phases



Building Blocks View of System Development



Scope Definition Phase

- **Problem statement** – a statement and categorization of problems, opportunities, and directives; may also include constraints and an initial vision for the solution. Synonyms include *preliminary study* and *feasibility assessment*.
- **Constraint** – any factor, limitation, or restraint that may limit a solution or the problem-solving process.
- **Scope creep** – a common phenomenon wherein the requirements and expectations of a project increase, often without regard to the impact on budget and schedule.
- **Statement of work** – a contract with management and the user community to develop or enhance an information system; defines vision, scope, constraints, high-level user requirements, schedule, and budget. Synonyms include *project charter*, *project plan*, and *service-level agreement*.

Requirements Analysis Phase

- What capabilities should the new system provide for its users?
- What data must be captured and stored?
- What performance level is expected?
- What are the priorities of the various requirements?

Logical Design Phase

Logical design – the translation of business user requirements into a system model that depicts only the business requirements and not any possible technical design or implementation of those requirements. Common synonyms include *conceptual design* and *essential design*.

System model – a picture of a system that represents reality or a desired reality. System models facilitate improved communication between system users, system analysts, system designers, and system builders.

Analysis paralysis –a common project condition in which excessive system modeling dramatically slows progress toward implementation of the intended system solution.

Decision Analysis Phase

- Candidate solutions evaluated in terms of:
  - **Technical feasibility** – Is the solution technically practical? Does our staff have the technical expertise to design and build this solution?
  - **Operational feasibility** – Will the solution fulfill the users' requirements? To what degree? How will the solution change the users' work environment? How do users feel about such a solution?
  - **Economic feasibility** – Is the solution cost-effective?
  - **Schedule feasibility** – Can the solution be designed and implemented within an acceptable time?
  - **Risk feasibility** – What is the probability of a successful implementation using the technology and approach?

Physical Design & Integration Phase

Physical design – the translation of business user requirements into a system model that depicts a technical implementation of the users' business requirements. Common synonyms include *technical design* or *implementation model*.

Two extreme philosophies of physical design

- *Design by specification* – physical system models and detailed specification are produced as a series of written (or computer-generated) blueprints for construction.
- *Design by prototyping* – Incomplete but functioning applications or subsystems (called *prototypes*) are constructed and refined based on feedback from users and other designers.

Construction and Testing Phase

- Construct and test system components
  - Software
    - Purchased
    - Custom-built
  - Databases
  - User and System Interfaces
  - Hardware
  - Networks

Installation and Delivery Phase

- Deliver the system into operation (production)
- Deliver User training
- Deliver completed documentation
- Convert existing data

System Operation & Maintenance

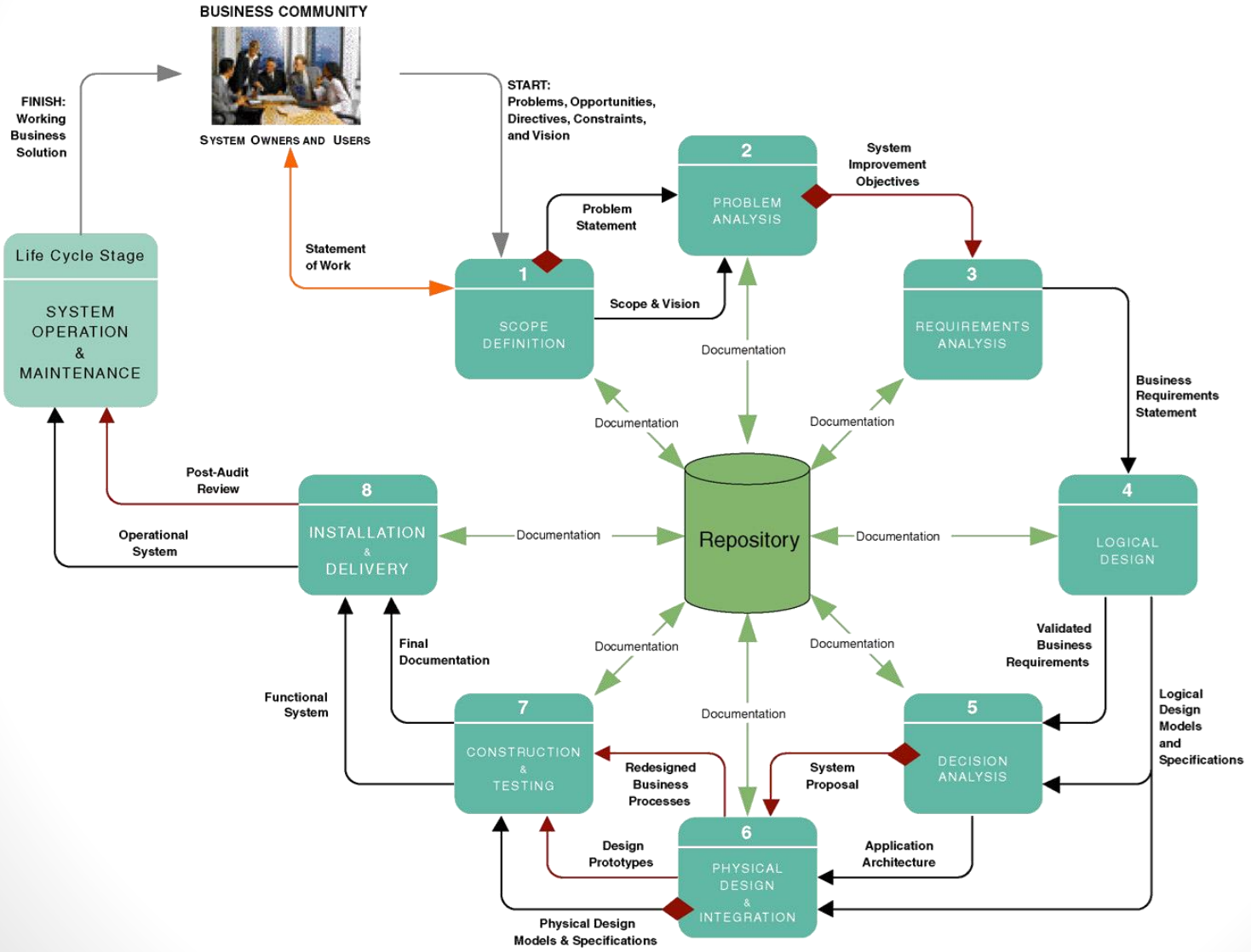
System support – the ongoing technical support for users of a system, as well as the maintenance required to deal with any errors, omissions, or new requirements that may arise.

Cross Life-Cycle Activities

Cross life-cycle activity – activities that overlap multiple phases

- **Fact-finding** - formal process of using research, interviews, meetings, questionnaires, sampling, and other techniques to collect information about system problems, requirements, and preferences.
- **Documentation and presentation**
  - **Documentation** – recording facts and specifications for a systems for current and future reference.
  - **Presentation** – communicating findings, recommendations, and documentation for review by interested users and mangers.
  - **Repository** – database and/or file directory where system developers store all documentation, knowledge, and artifacts for information systems or project(s)
- **Feasibility analysis**
- **Process and project management**

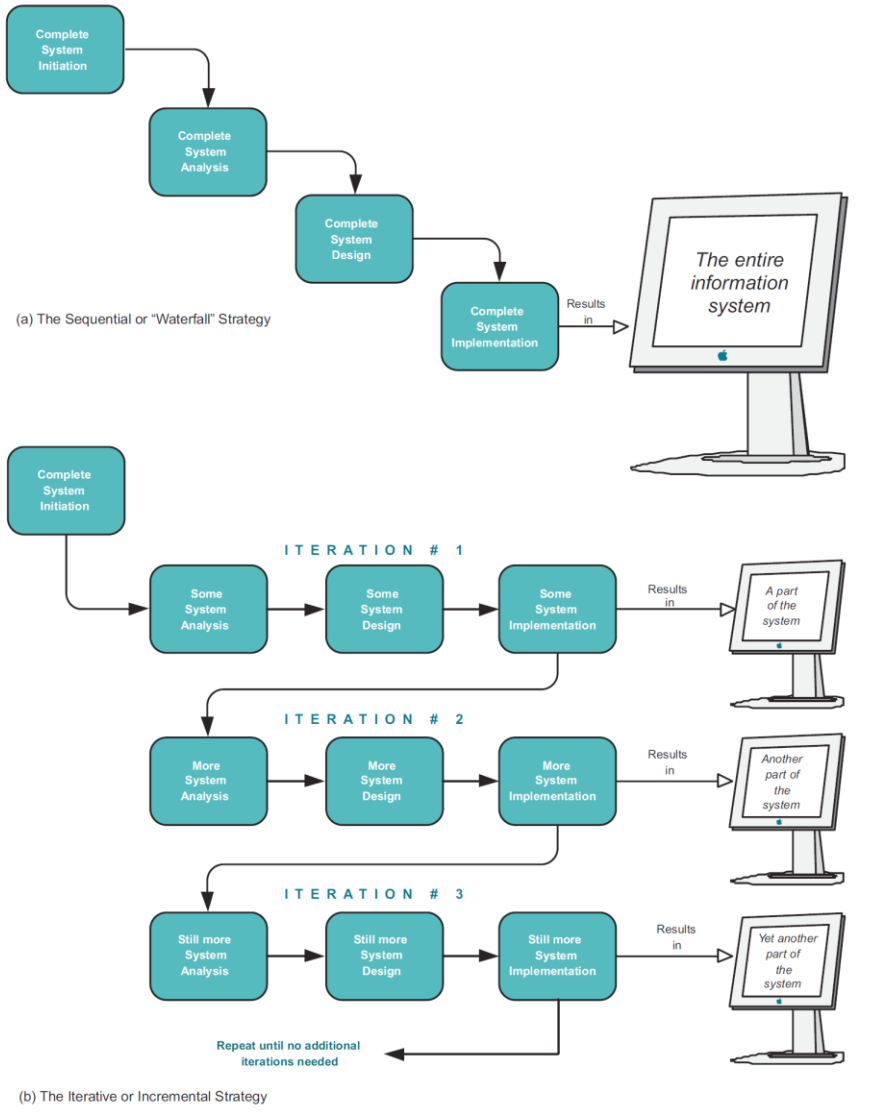
System Development Documentation, Repository, and Presentations



Sequential versus Iterative Development

Waterfall development approach an approach to systems analysis and design that completes each phase one after another and only once .

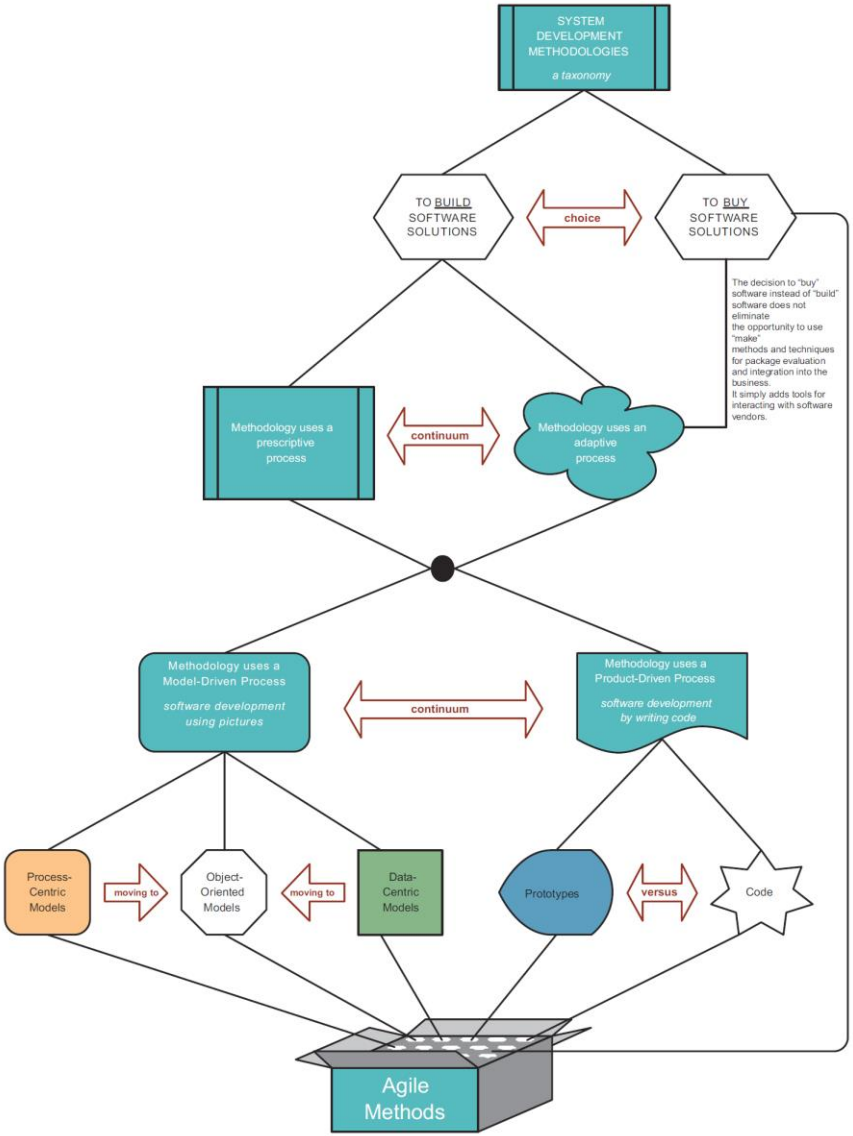
Iterative development approach an approach to systems analysis and design that completes the entire information system in successive iterations. Each iterations does some analysis, some design, and some construction. Synonyms include incremental and spiral.



Alternative Routes and Strategies

- Methodologies and routes can support the option of either *building* software solutions in-house or *buying* a commercial software solution from a software vendor.
- Methodologies may be either very *prescriptive* (“Touch all the bases; follow all the rules”) or relatively *adaptive* (“Change as needed within certain guidelines”).
- Methodologies can also be characterized as *model-driven* (“Draw pictures of the system”) or *product-driven* (“Build the product and see how the users react”).
- Model-driven methodologies have moved to a focus on the object oriented technologies. Earlier model-driven approaches emphasized either process modeling or data modeling.
- Product-driven approaches tend to emphasize either rapid prototyping or writing program code as soon as possible (extreme programming).

A Taxonomy for System Development Methodologies & Strategies

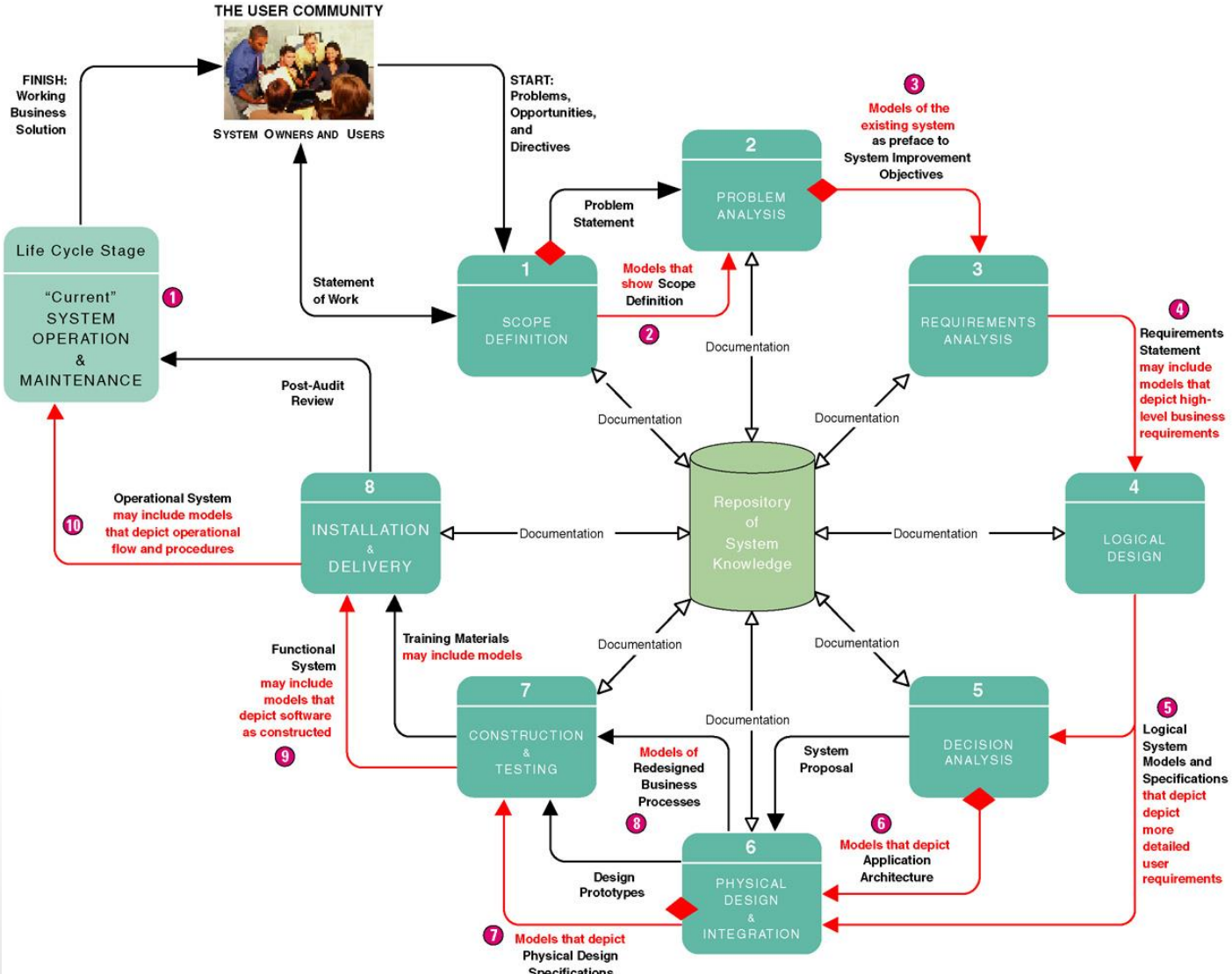


Logical vs. Physical Models

Logical model - a pictorial representation that depicts what a system is or does.

Physical model - a technical pictorial representation that depicts what a system is or does and how the system is implemented.

Model-Driven Development Strategy



Model-Driven Development Strategy

- **Model-driven development** – a system development strategy that emphasizes the drawing of system models to help visualize and analyze problems, define business requirements, and design information systems.
  - **Process modeling** – a process-centered technique popularized by the structured analysis and design methodology that used models of business process requirements to derive effective software designs for a system.
  - **Data modeling** – a data-centered technique used to model business data requirements and design database systems that fulfill those requirements.
  - **Object modeling** – a technique that attempts to merge the data and process concerns into singular constructs called objects. Object models are diagrams that document a system in terms of its objects and their interactions.

Model-Driven Development Strategy

Advantages

- Requirements often more thorough
- Easier to validate business requirements
- Easier to analyze alternatives
- Design specifications often more stable and flexible
- Systems can be constructed more correctly the first time

Model-Driven Development Strategy

Advantages

- Requirements often more thorough
- Easier to validate business requirements
- Easier to analyze alternatives
- Design specifications often more stable and flexible
- Systems can be constructed more correctly the first time

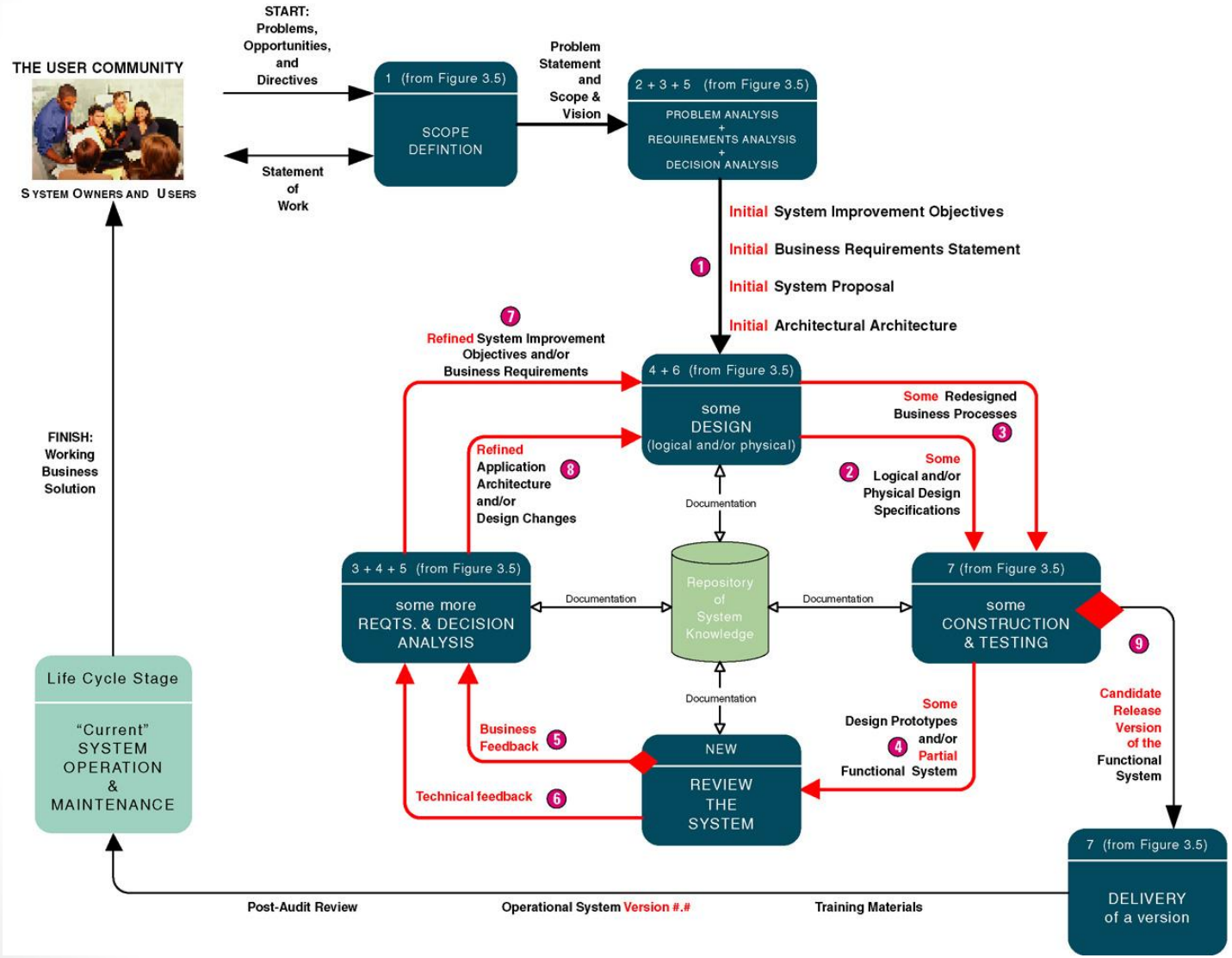
Disadvantages

- Time consuming
- Models only as good as users' understanding of requirements
- Reduces users' role because pictures are not software
- Can be Inflexible

Rapid Application Development Strategy

- **Rapid application development** (RAD) – a system development strategy that emphasizes speed of development through extensive user involvement in the rapid, iterative, and incremental construction of series of functioning prototypes of a system that eventually evolves into the final system.
  - **Prototype** – a small-scale, representative, or working model of the users' requirements or a proposed design for an information system.
  - **Time box** – the imposition of a non-extendable period of time, usually 60-90 days, by which the first (or next) version of a system must be delivered into operation.

Rapid Application Development Strategy



Rapid Application Development Strategy

Advantages

- User requirements often uncertain or imprecise
- Encourages active user and management participation
- Projects get higher visibility and support
- Stakeholders see working solutions more rapidly
- Errors detected earlier
- Testing and training are natural by-products
- More natural process because change is expected

Rapid Application Development Strategy

Advantages

- User requirements often uncertain or imprecise
- Encourages active user and management participation
- Projects get higher visibility and support
- Stakeholders see working solutions more rapidly
- Errors detected earlier
- Testing and training are natural by-products
- More natural process because change is expected

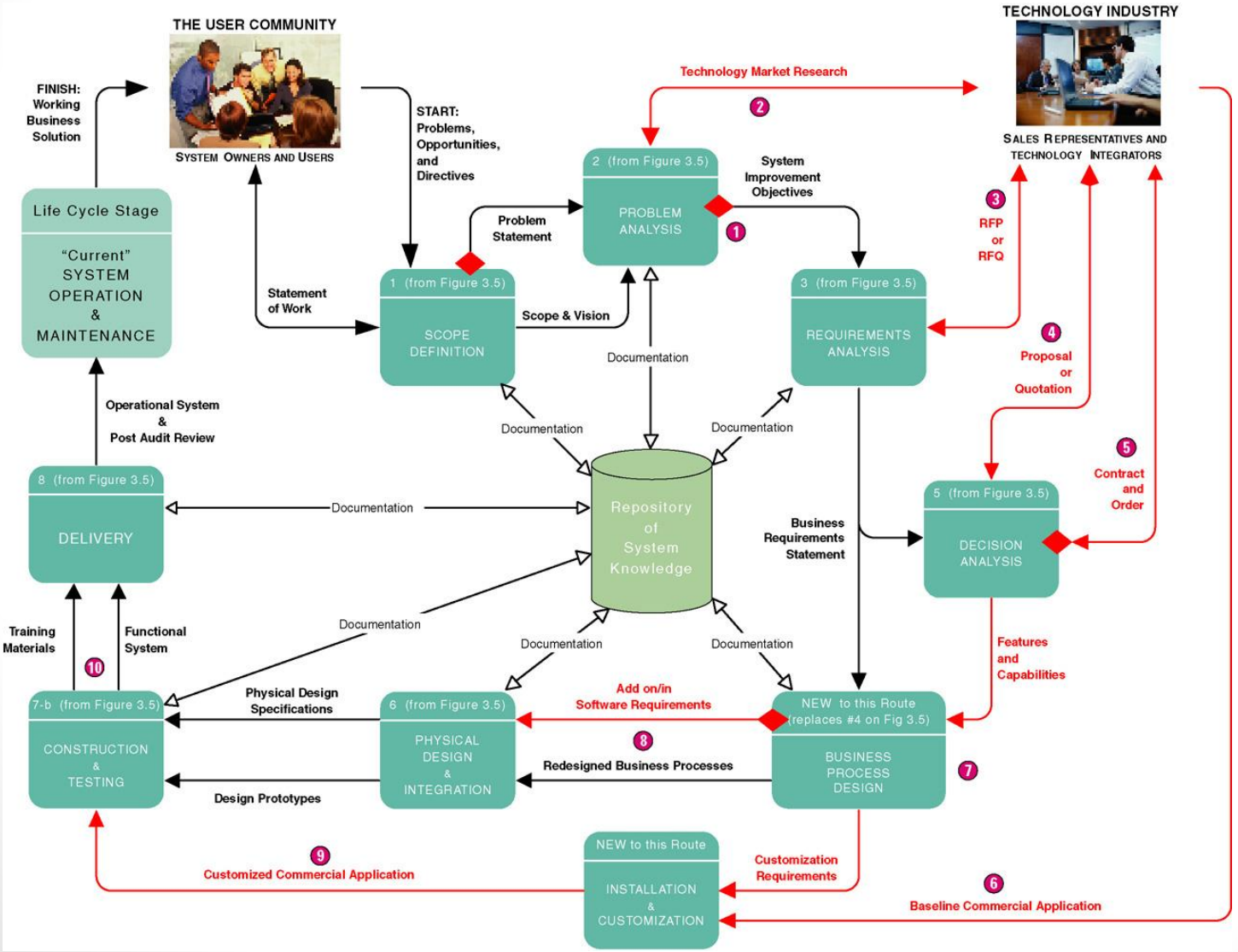
Disadvantages

- May encourage "code, implement, repair" mentality
- Can solve wrong problem since problem analysis is abbreviated
- May discourage analysts from considering alternatives
- Stakeholders reluctant to throw away prototype
- Emphasis on speed can adversely impact quality

Commercial Application Package Implementation Strategy

- **Commercial application package** – software application that can be purchased and customized to meet business requirements of a large number of organizations or specific industry. A synonym is *commercial off-the-shelf* (COTS) system.
  - **Request for proposal (RFP)** – formal document that communicates business, technical, and support requirements for application software package to vendors that may wish to compete for the sale of application package and services.
  - **Request for quotation (RFQ)** – formal document that communicates business, technical, and support requirements for an application software package to a single vendor that has been determined as being able to supply that application package and services.
  - **Gap analysis** – comparison of business and technical requirements for a commercial application package against capabilities and features of a specific commercial application package to define requirements that cannot be met.

Commercial Application Package Implementation Strategy



Commercial Application Package Implementation Strategy

Advantages

- Systems usually implemented more quickly
- Avoids staffing required to develop in-house solutions
- Generally less expensive
- Vendor assumes responsibility for improvements and corrections
- Many business functions more similar than dissimilar for all businesses in a given industry

Commercial Application Package Implementation Strategy

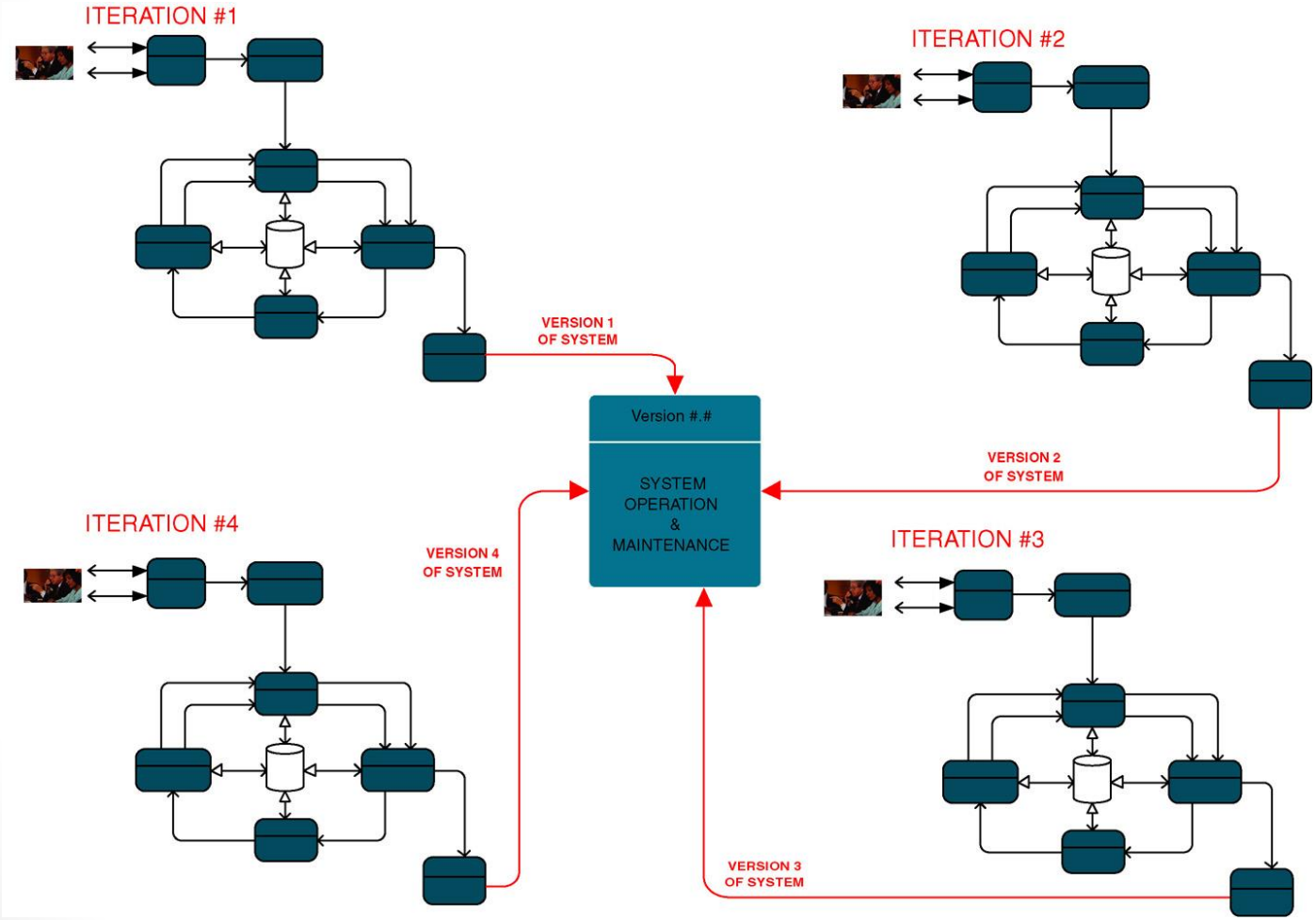
Advantages

- Systems usually implemented more quickly
- Avoids staffing required to develop in-house solutions
- Generally less expensive
- Vendor assumes responsibility for improvements and corrections
- Many business functions more similar than dissimilar for all businesses in a given industry

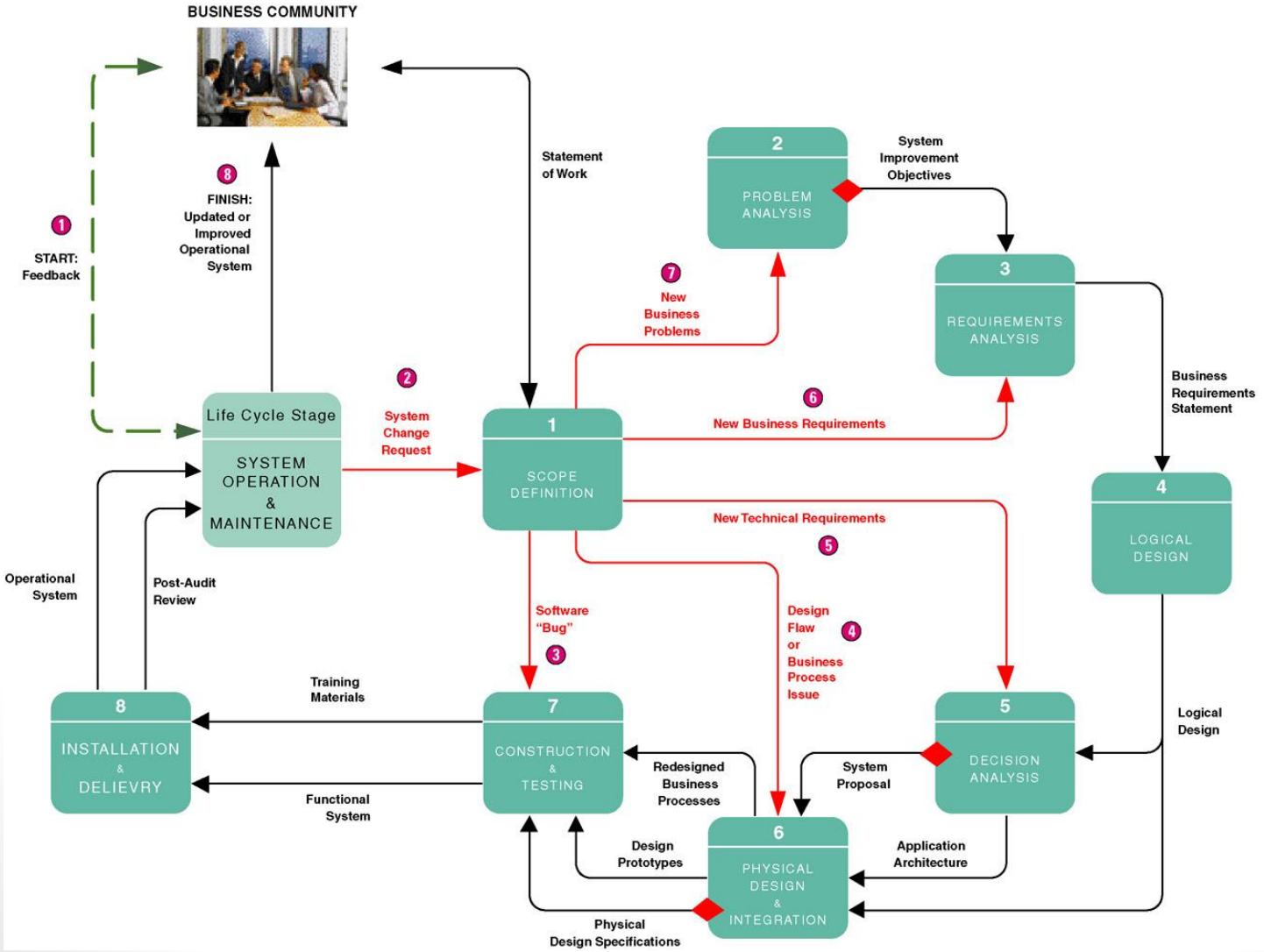
Disadvantages

- Dependent on long-term viability of vendor
- Rarely reflects ideal solution
- Often resistance to changes business processes to adapt to software

Hybrid Strategies



A System Maintenance Perspective



Automated Tools and Technology

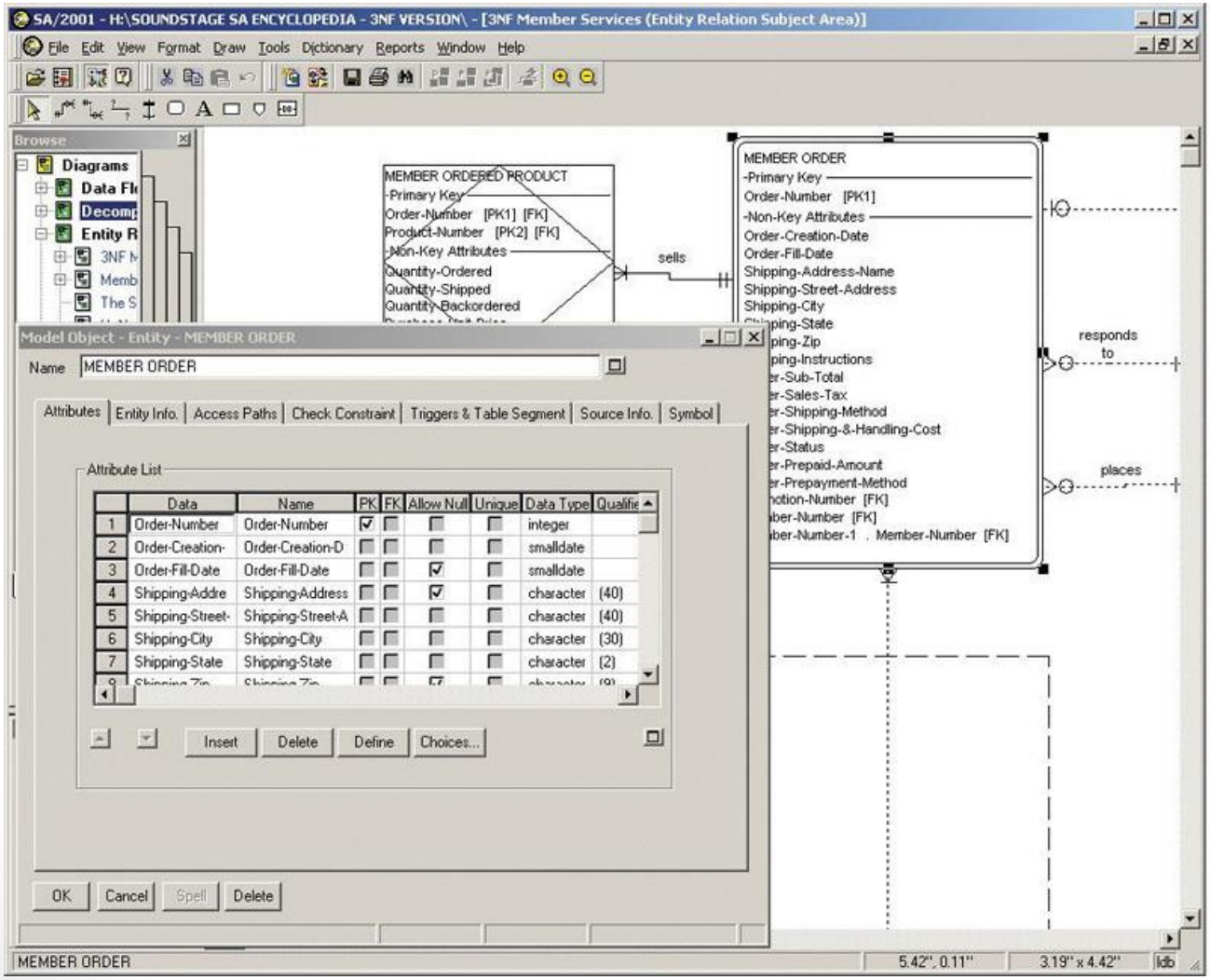
- Computer-aided systems engineering (CASE)
- Application development environments (ADEs)
- Process and project managers

Computer-Assisted Software Engineering (CASE)

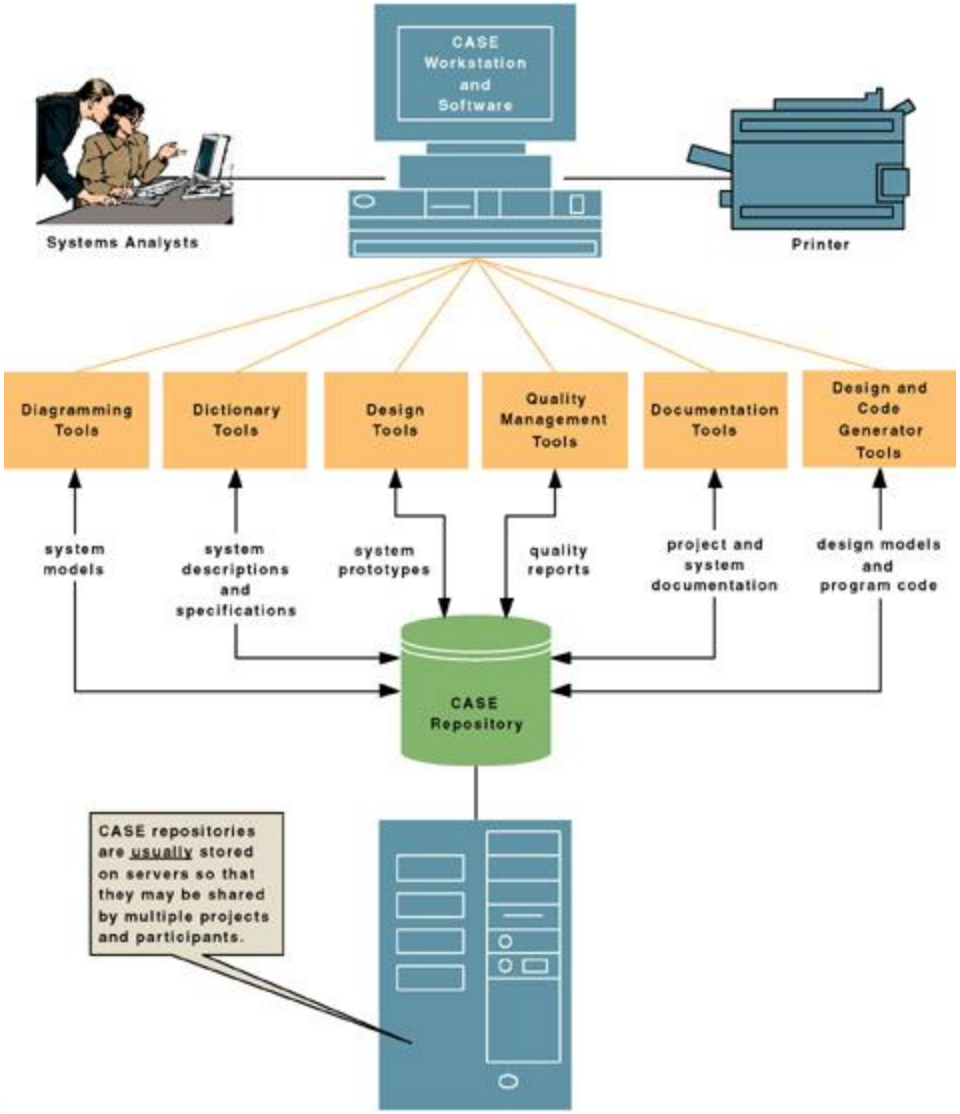
Computer-aided systems engineering (CASE) –automated software tools that support the drawing and analysis of system models and associated specifications. Some CASE tools also provide prototyping and code generation capabilities.

- **CASE repository** – system developers' database where developers can store system models, detailed descriptions and specifications, and other products of system development. Synonyms: *dictionary* and *encyclopedia*.
- **Forward engineering** – CASE tool capability that can generate initial software or database code directly from system.
- **Reverse engineering** – CASE tool capability that can generate initial system models from software or database code.

Using a CASE Tool for System Development



CASE Tool Architecture



Application Development Environments

Application development environments (ADEs) – an integrated software development tool that provides all the facilities necessary to develop new application software with maximum speed and quality. A common synonym is *integrated development environment* (IDE)

- ADE facilities may include:
  - Programming languages or interpreters
  - Interface construction tools
  - Middleware
  - Testing tools
  - Version control tools
  - Help authoring tools
  - Repository links

Process and Project Managers

- **Process manager application** – an automated tool that helps document and manage a methodology and routes, its deliverables, and quality management standards. An emerging synonym is *methodware*.

- **Project manager application** – an automated tool to help plan system development activities (preferably using the approved methodology), estimate and assign resources (including people and costs), schedule activities and resources, monitor progress against schedule and budget, control and modify schedule and resources, and report project progress.

Any Questions?

Without requirements and design, programming is the art of adding bugs to an empty text file.

Louis Srygley