Systems Analysis and Design in an Age of Options

G. Spurrier and H. Topi

Chapter 1—Overview and Key Concepts

Chapter 1

Systems Analysis and Design in an Age of Options

Learning Objectives

- ***Explain why software development is a highly labor-intensive process that requires aptitudes and skill sets that only human beings—not computers—possess, including:***
  - ***Creativity***
  - ***Complex communications***
  - ***Large-frame pattern recognition***
  - ***Asking interesting questions***
  - ***Common sense***
- ***Express the core responsibility of the business analyst to engage in systems analysis, including:***
  - ***Determining features—what the system must do***
  - ***Design—how the system should do it***
- ***Understand that the business analyst cannot merely be a passive note taker, but rather must actively help business customers drive the systems project through innovation, deep understanding, and good judgment.***

Learning Objectives

- ***Explain the key challenge of systems analysis and design in the current age of options of choosing between two fundamental systems project options:***
  - ***Plan-driven: Where the project is like building a house, in which we can create detailed, up-front blueprints that the team then constructs.***
  - ***Agile: Where the project is like inventing a gadget, in which what needs to be built must emerge from constructing a series of prototypes.***
- ***Utilize the Systems Development Process Framework to understand project tasks from the beginning to the end of a systems project.***
- ***Realize how the impact of agile has extended the role of the business analyst to include project leadership throughout the systems project.***
- ***Appreciate how the business analyst must confront and overcome a variety of ethical challenges.***

Chapter Outline

- Introduction
- The Core Role of the BA: Software Requirements Analysis
- Beyond SA&D Requirements Analysis: Understanding the Big Picture of System Projects in the Plan-Driven Approach
- A Major Alternative to Plan-Driven: Agile Approaches
- The BA in an Age of Options
- Security: A Critically Important Topic That Involves Every Team Member
- Ongoing Case Study and Minicases
- Summary

Introduction

Using Software: Focus on Technology

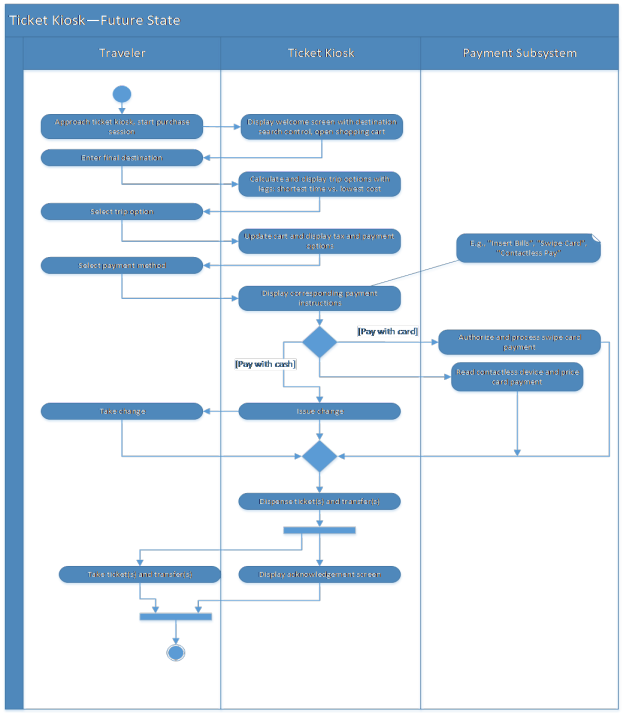


Figure 1-1 Using software: automated transactions, management information, and disruptive technologies (From iStock.com/enotmaks; iStock.com/metamorworks)

Creating Software: Focus on Human Teams

- ***Creativity***
- ***Complex communications***
- ***Large-frame pattern recognition***
- ***Asking interesting questions***
- ***Common sense***

Creating Software: A Human-Driven Process



Figure 1-2. Creating software: requirements, programming, and implementation
(From iStock.com/SeventyFour; iStock.com/skynesher; iStock.com/dima_sidelnikov)

Core Role of the BA: Software Requirements Analysis

Software Projects: Difficult and Risky

- Standish Group Report (1995):
  - 16% completely successful
  - 53% failed to fully meet expectations
  - 31% cancelled prior to delivery
- Why is this so?

Why So Difficult? The Obvious: Software Construction

```
package fileiowithlooping;
import java.util.Scanner;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.text.NumberFormat;
import java.util.Locale;

public class FileIOwithLooping {
   public static void main(String[] args) {
      Scanner fileIn = null;
      try {
         /* Next line opens the file -- note use of escape characters */
         fileIn = new Scanner(new FileInputStream
            ("C:\\Users\\gspurrier\\Documents\\Products.txt"));
      }
      /* Following code executes if file is not found, including exit */
      catch (FileNotFoundException e) {
         System.out.println("File not found.");
         System.exit(0);
      }
      /* Program only reaches this point if file is opened successfully */
      NumberFormat  money = NumberFormat.getCurrencyInstance(Locale.US);
      String code = "";
      String description = "";
      double price = 0.0;
      int inventory = 0;
      double inventoryValue = 0.0;
      double totalInvValue = 0.0;
      System.out.println("Code\tDescrip\tPrice\tInv.\tValue");
      System.out.println("-----------------------------------------");
      String line = null;
      while (fileIn.hasNextLine())
      {
         line = fileIn.nextLine();
         String[] columns = line.split(",");
         code = columns[0];
         description = columns[1];
         price = Double.parseDouble(columns[2]);
         inventory = Integer.parseInt(columns[3]);
         inventoryValue = price * inventory;
         totalInvValue += inventoryValue;
         System.out.println(code + "\t" + description + "\t" +
               money.format(price) + "\t" + inventory + "\t" +
               money.format(inventoryValue));
      }
      System.out.print("Total Inventory Value = ");
      System.out.println(money.format(totalInvValue));
   }

}
```

Figure 1-3 A simple Java program—like reading a foreign language

Why So Difficult? Off-the-Mark Software Requirements

- Poor software construction = not running as designed
- Poor software requirements =
  - Wrong features: *What* the system needs to do
  - Wrong design: *How* the system will deliver those features
    - Data
    - Logic
    - User Interface

Core Role of the BA: Software Requirements

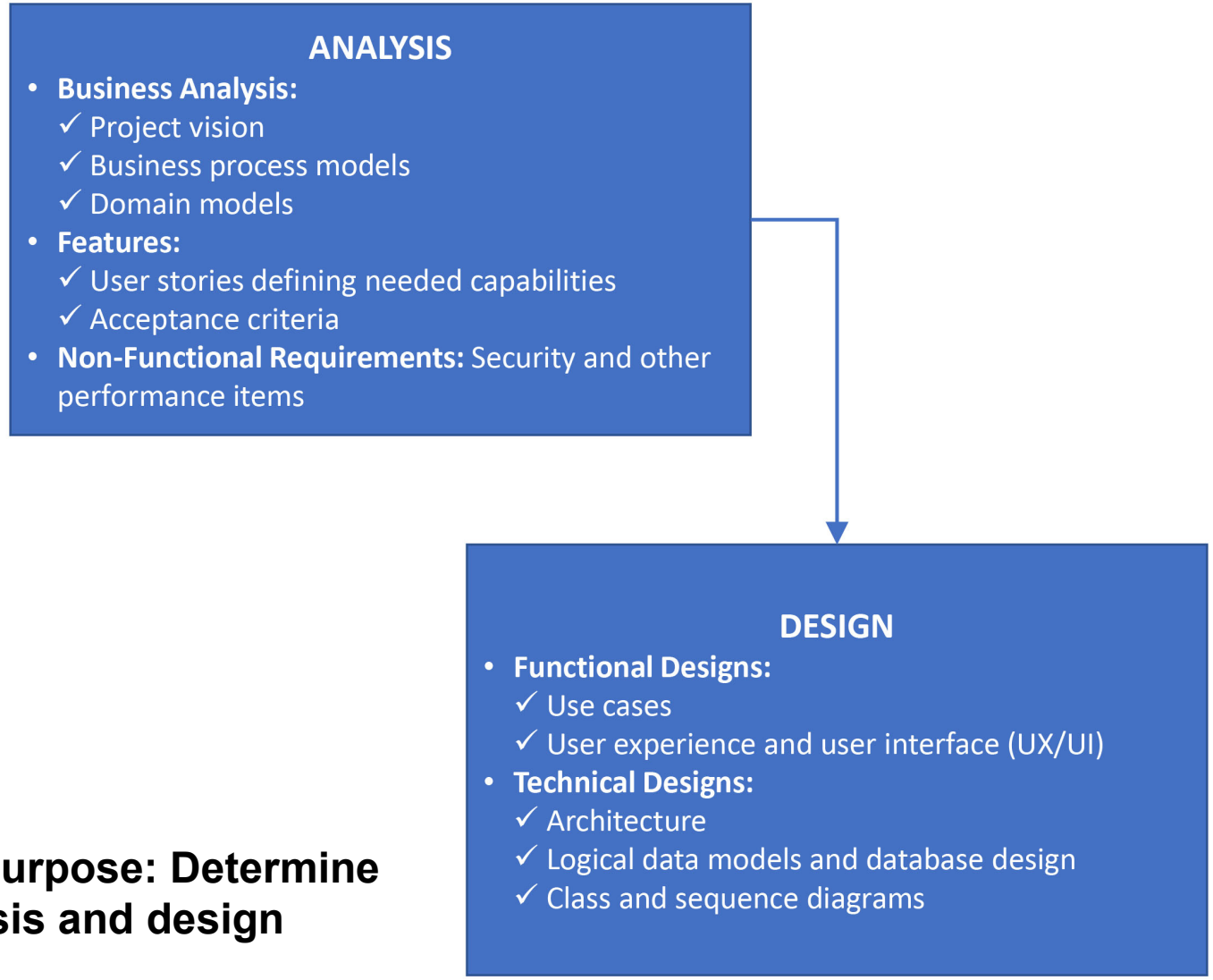
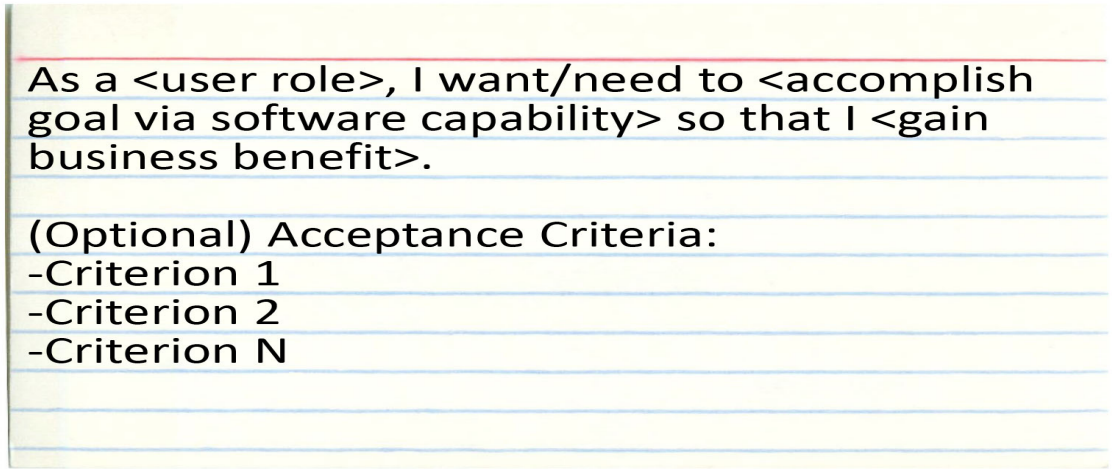


Figure 1-7 SA&D core purpose: Determine requirements via analysis and design

Specifying Software Features with User Stories



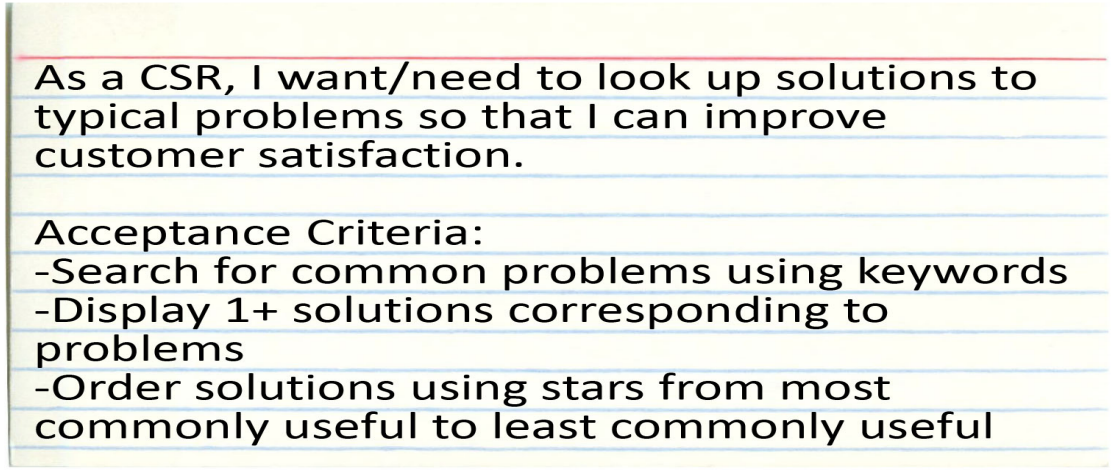


Figure 1-6 The format and a specific example of a user story and optional supporting acceptance criteria, illustrating how it can fit on a three-by-five-inch card

User Stories: Example

- User Story:
  - Format: "As a *type of customer*, I want/need *some kind of feature* so that I can *obtain some goal or benefit*."
  - Information: Who, What, and Why (but NOT How)
- Examples: For a motor vehicle
  - *As a race car driver, I want a car that can accelerate rapidly so that I can pass other drivers.*
  - *As a race car driver, I want a car that can attain extremely high top speeds so that I can stay ahead of other drivers.*
  - *As a race car crew chief, I want a car that enables me to change the tires in under 12 seconds so that we don't lose our position during pit stops.*

Differing Feature Requirements

Figure 1-4 Two Ford Motor Company vehicles satisfying different sets of feature requirements
(From iStock.com/dima_sidelnikov; iStock.com/contrastaddict)

Differing Design Requirements

- How the solution will deliver the features:
  - A race car that resembles a regular sedan, is optimized to race around a paved oval making only left-hand turns, and is limited to certain engine sizes and other specifications to make the racing more competitive based on driver skills.
  - A race car focused on road racing on complicated, winding race tracks making both left-hand and right-hand turns, and using highly streamlined body shape and airfoils to create downforce to maximize the performance of each individual car.

Differing Design Requirements

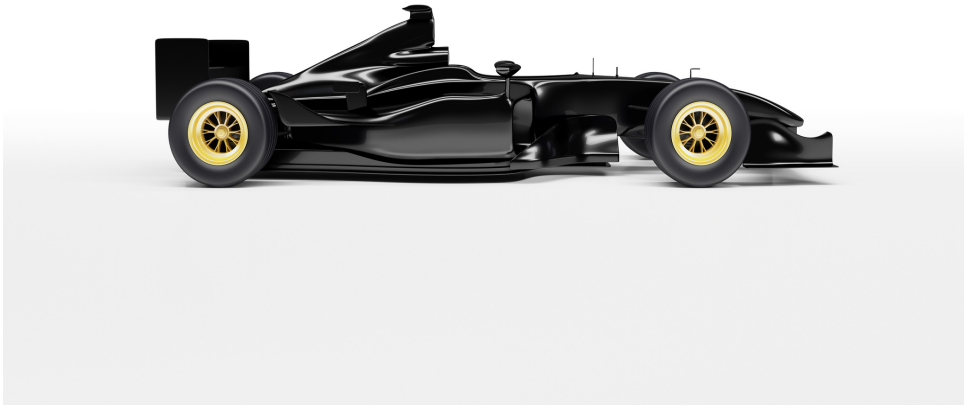


Figure 1-5 Two race cars satisfying different design requirements: a NASCAR stock car versus a Formula 1 car (From iStock.com/avid_creative; iStock.com/chromatika)

Implementation Options: Construction vs. Configuration

- **Construction:**
  - Traditional approach
  - Programming new features (e.g. Java, C#, or Python)
- **Configuration:**
  - Increasingly important alternative
  - Selecting and configuring third-party software
  - COTS (commercial-off-the-shelf) software

Why SA&D is So Challenging

- ***The BA:***
  - ***Cannot merely be a passive note taker***
  - ***Must actively help business customers drive the systems project through***
    - ✓ ***Innovation***
    - ✓ ***Deep understanding***
    - ✓ ***Good judgment***

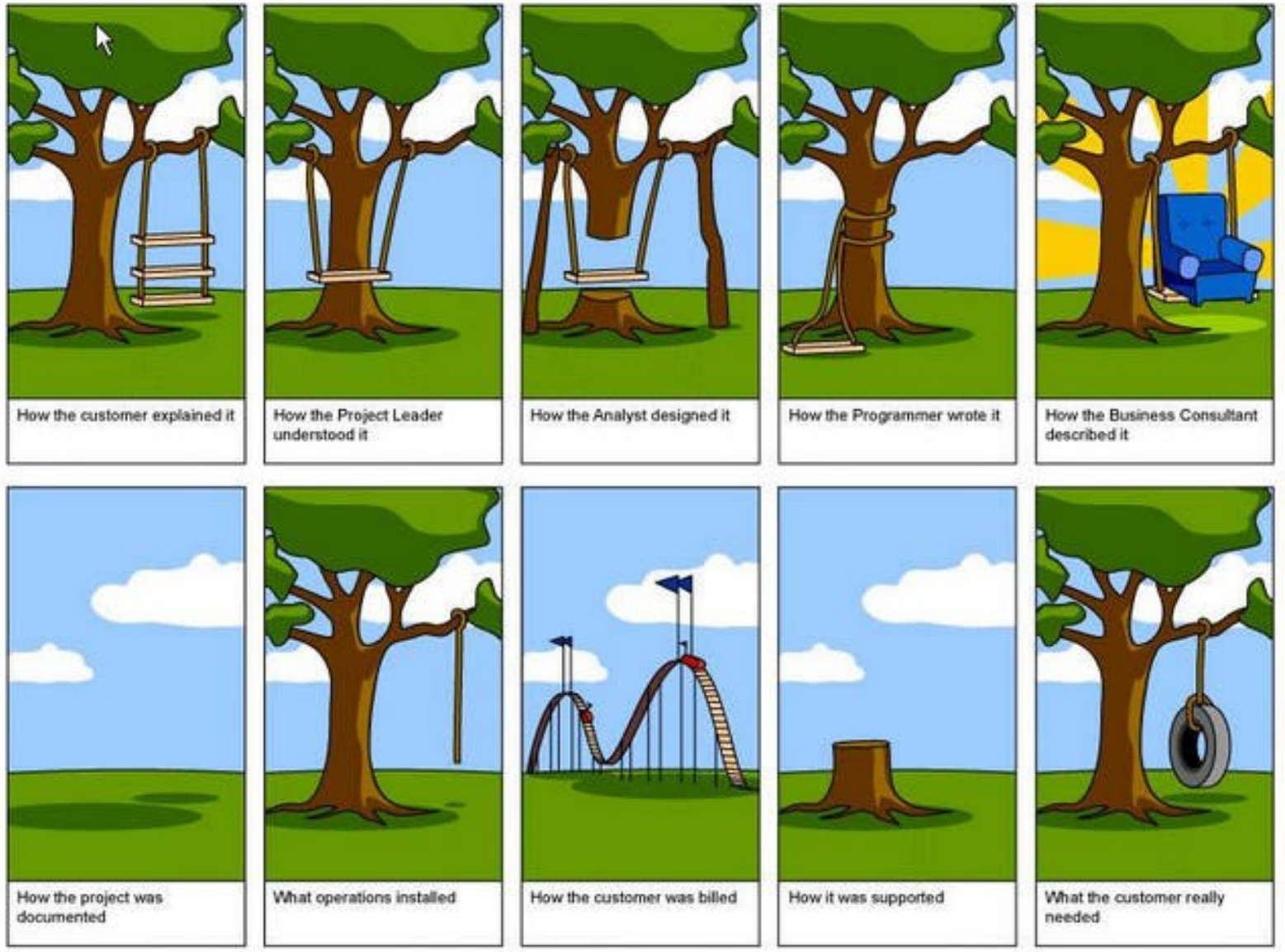


Figure 1-8 A metaphor for the many points where the systems requirements and implementation process can go wrong (copyright © projectcartoon.com under the Creative Commons Attribution 3.0 License)

Beyond SA&D Requirements Analysis: Understanding the Big Picture of Systems Projects in the Plan-Driven Approach

Plan-Driven Approach: Like Building a House

Figure 1-9 Using detailed blueprints to plan the construction of a house
(From iStock.com/Kwangmoozaa)

Plan-Driven Approach

- Traditional approach
  - "Traditional SDLC"
  - "Waterfall"
- Key activities executed one after another in a linear approach
- Big Requirements Up Front (BRUF)

Plan-Driven Approach

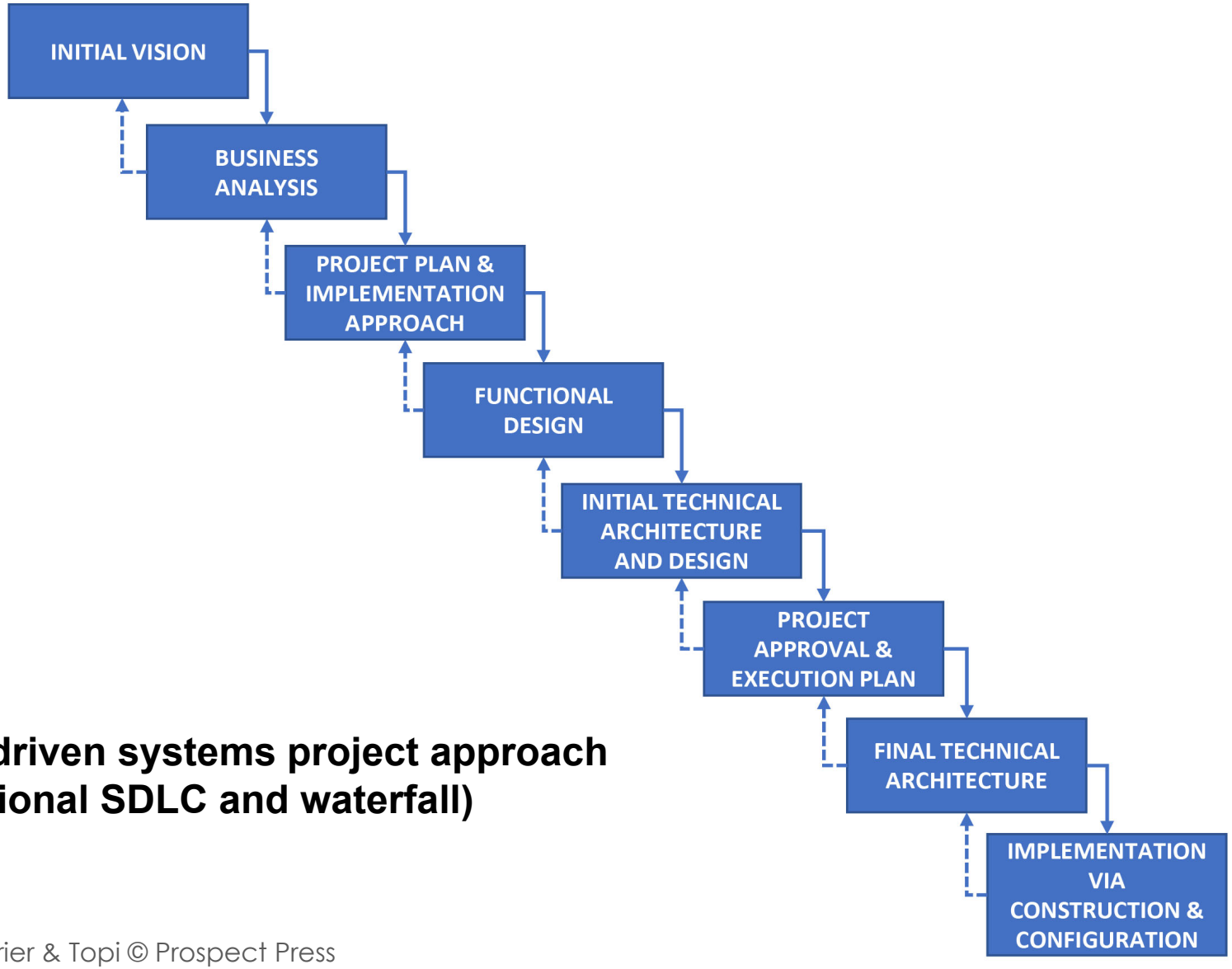


Figure 1-10 The plan-driven systems project approach (which includes traditional SDLC and waterfall)

Key System Categories

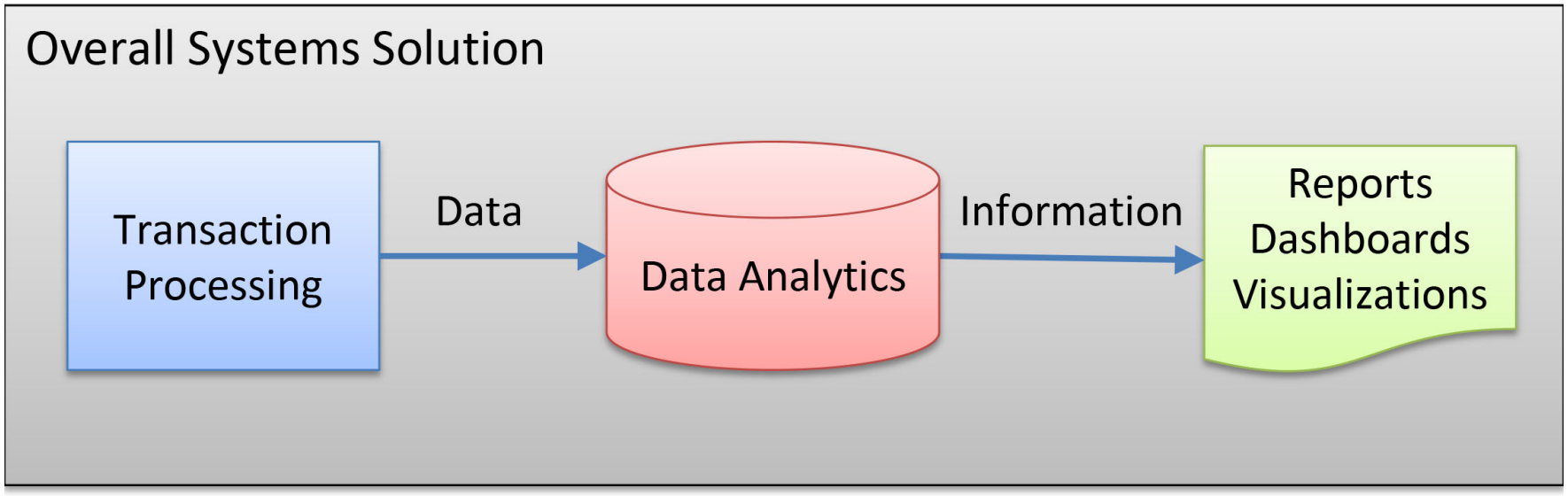


Figure 1-11 Combination of transaction processing and data analytics system types to provide overall systems support

Business Analysis: Modeling Current vs. Future State

- Current State:
  - Way organization currently operates
  - Identifying problems or opportunities that IT can address
- Future State:
  - Way organization will operate with new/improved software
- Requirements:
  - New Capabilities =
    Future State Capabilities
    – Current State Capabilities
    + Refactoring

Functional vs. Technical Designs

- **Functional Design:** How the system works from customer perspective
  - Logic
  - Data
  - User Interface
- **Technical Designs & Architecture:** Focus on underlying technology
  - Technical Design: Translate functional designs to programming components
  - Architecture: Overall approach to data storage, processing, communications

Major Alternative to Plan-Driven: Agile Approaches

Agile Approaches

- Motivated by poor project outcomes through 1990s
- Examples:
  - eXtreme Programming
  - Scrum
- Characteristics:
  - Iterative construction (“sprints”)
  - Emergent requirements

Agile Assumptions

- Software development:
  - More like inventing a gadget than building a house
  - Big Requirements Up Front not a viable option
  - So instead "Build a little, review a little, revise a little"

Agile as a Metaphor for Continuous (Re)Invention

Figure 1-13 Continuous (re)invention of the airplane (from iStock.com/Kwangmoozaa)

Agile Approach

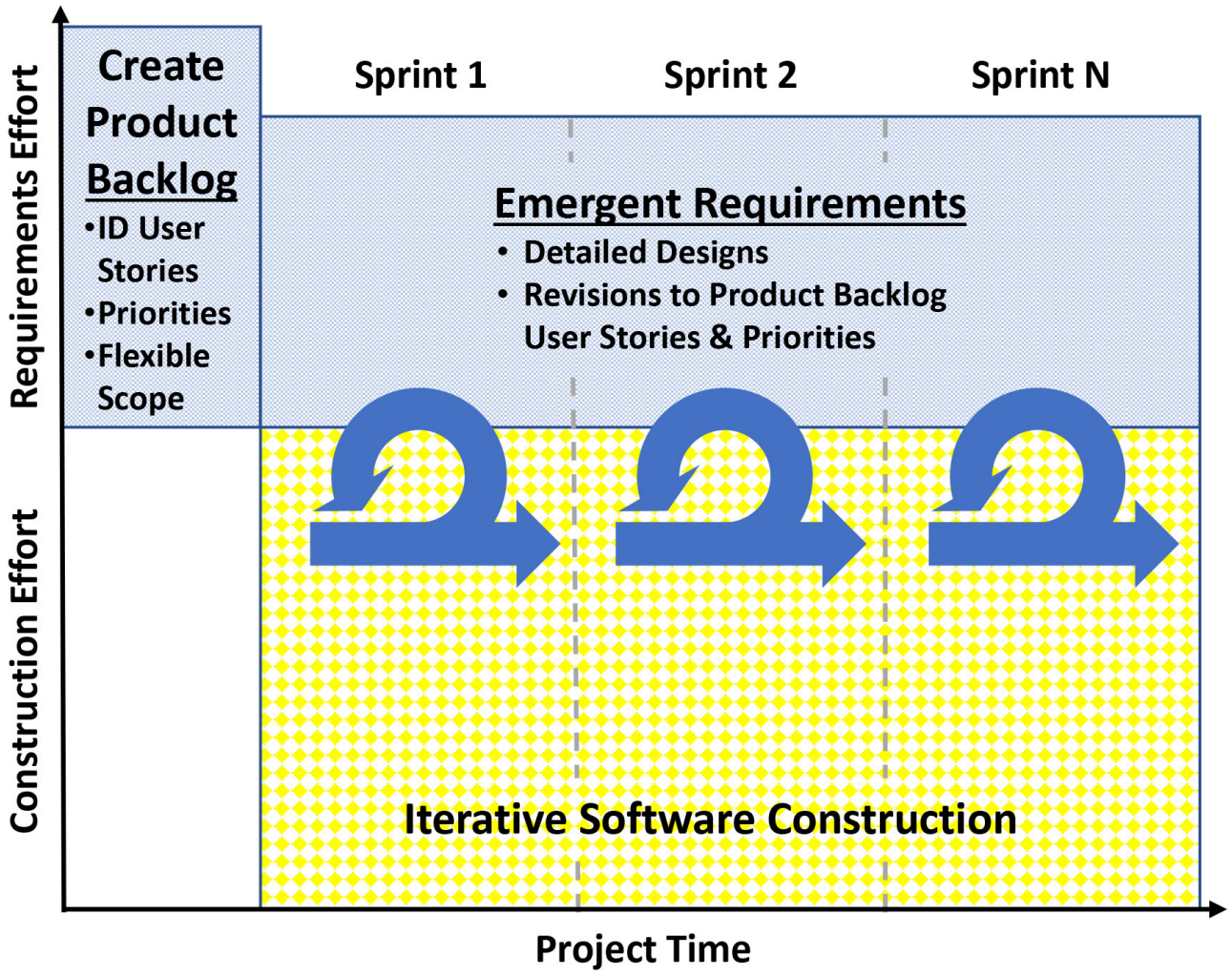


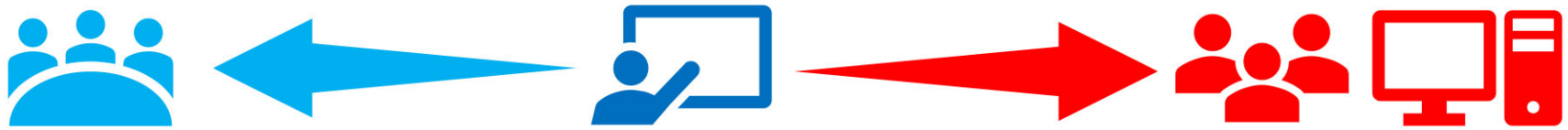
Figure 1-12 Agile approach to systems projects

One More Major Option

- Hybrid:
  - A strong, modern approach for large, complex projects
  - Combination of:
    - Big Requirements Up Front
    - Iterative Construction (with some re-planning of requirements)

The BA in an Age of Options …

BA as the Bridge between Business and IT



Business Team:

- Leadership
- Finance
- Accounting
- Marketing
- Operations
- Manufacturing
- Logistics
- Human Resources
- Etc.

Business/Systems Analyst:

1. Envisioning
2. Business Analysis
3. Project Planning & Implementation Approach Selection
4. Functional and Technical Design
5. Project Approval and Execution Planning
6. Implementation via Construction or Configuration
7. Deployment and Maintenance

Technologists:

- Developers
- Software Engineers
- Enterprise Architects
- Networking Specialists
- Cyber Security Specialists
- Etc.

Figure 1-14 Role of the business analyst as a bridge between the business team and IT

Beyond Requirements: The Expanded Role of the BA

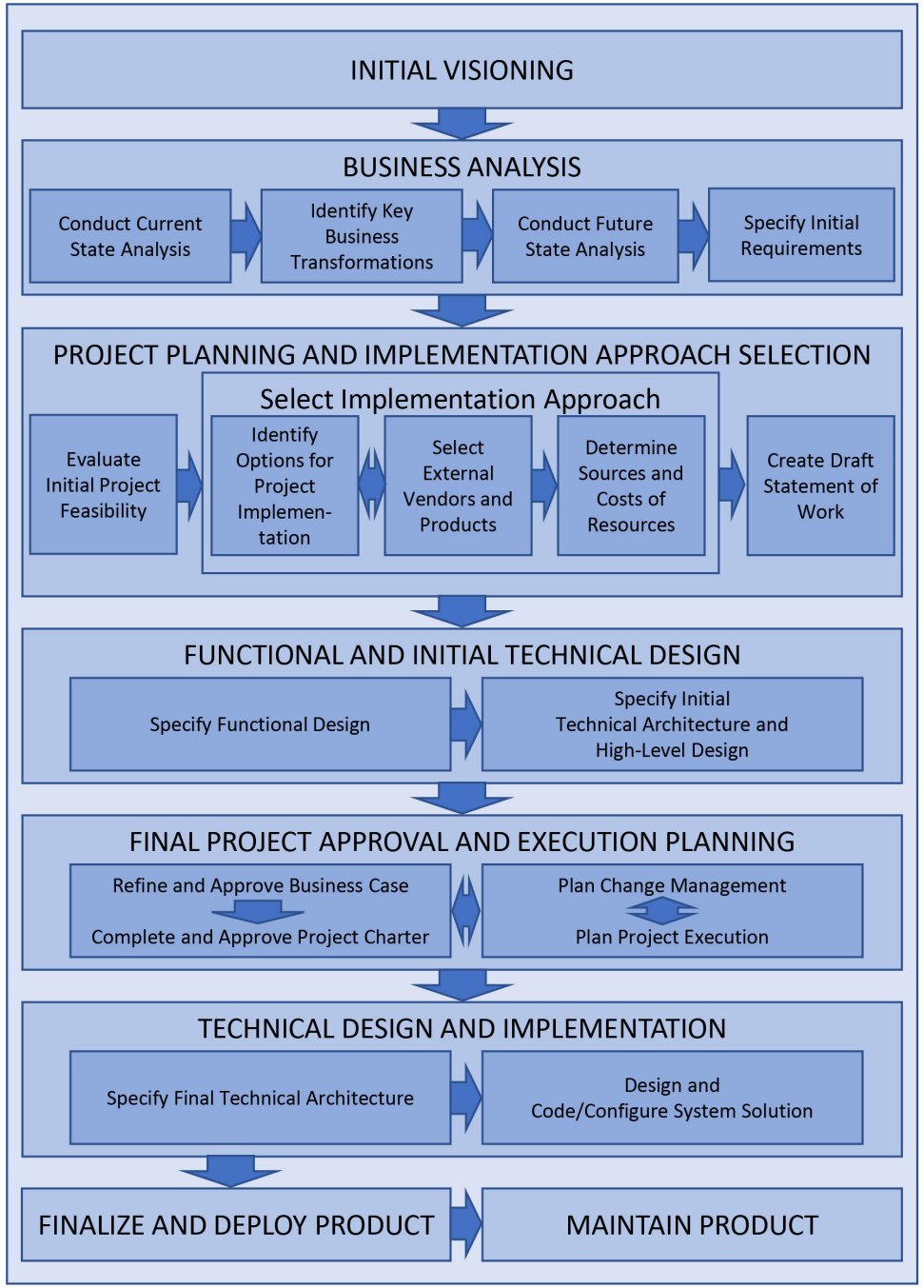
- BA stays engaged throughout project:
  - Envisioning through Change Management
  - Earliest and most visible contact with customer
- Expanded role of BA today:
  - Agile promotes small teams of generalists
  - BA therefore make take on wider roles
    - Project approach selection and planning
    - Project management
    - ScrumMaster
    - Testing
    - Documentation

System Development Process Framework

- Throughout, will relate overall BA activities to an organizing process framework
- Variations will include:
  - Agile
  - Hybrid

Figure 1-15 Systems Development Process Framework (plan-driven version)

Systems Development Process Framework: Plan-Driven Approach



Security: A Critically Important Topic That Involves Every Team Member

Cyber Security: An Increasing Concern

Driven by:

- Increasing amounts of sensitive, confidential data
- Systems accessible via the Internet
- Risk of hackers:
  - External
  - Internal

Chapter Summary

Let's Review

- Core BA role is to determine requirements: features and designs.
- Key systems project activities from the Systems Development Process Framework must be addressed in all systems projects
- Today the BA often performs an expanded range of roles in all project activities, including project management and many others.
- The BA today operates in an Age of Options: There is no single "best approach" to organize and execute software projects.
- Key approaches include:
  - Plan-driven: Projects can be comprehensively planned using BRUF and then constructed in a linear fashion.
  - Agile: Projects need emergent requirements during iterative construction.
  - Hybrid: Combines BRUF with iterative construction.
- Security has become a top priority because of high levels of sensitive data, Internet-accessible systems, and the rise of hackers.