Understanding and Modeling Organizational Systems

CSE 4407

Md. Bakhtiar Hasan

Assistant Professor
Department of Computer Science and Engineering
Islamic University of Technology

May 14, 2025

What is an Organization... as a System?

- Core Idea: Organizations are systems designed to achieve goals using resources (people, tech, etc.)
- Structure: Composed of smaller, interrelated **subsystems** (departments, units)
  - Example: Accounting, Marketing, Production, IT, Operations, Legal
- Integration: Subsystems work together to form the whole organization
- Key Benefit: Systems principles help us understand how organizations *really* work

The Web of Connections: Interrelatedness

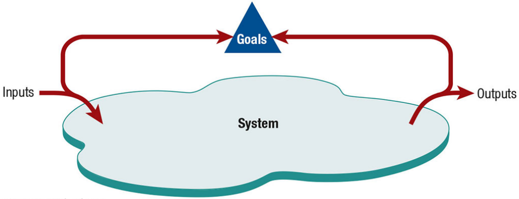
- Fundamental Truth: All systems and subsystems are **interrelated** and **interdependent**
- Ripple Effect: Changing one element affects the *entire* system
  - Removing admin assistants and replacing them with PCs impacts
    - The assistants (obviously!)
    - Managers (new tasks?)
    - Anyone who relied on the assistants' informal communication networks
- The takeaway for analysts: Always consider the wider impact of system changes

System Mechanics: Inputs, Processes, Outputs, and Boundaries

- Input: Resources from the environment (people, materials, information)
- Process: Transformation of inputs (verifying, updating, calculating, manufacturing)
  - *Is something being changed? If not, it might not be a process*
- Output: Results returned to the environments (products, services, information)
- Boundaries: Separate the system from its environment
  - Can be **permeable** (open, easy exchange) or **impermeable** (closed, restricted exchange)
  - Organizations need *some* permeability to survive (import resources, export products)

Staying on Course: Feedback and Environment

- System Control: Organizations use planning and control
- Feedback: Output is compared to goals to guide future inputs/actions
  - Example: Poor sales of red, white, and blue weight sets (output) leads to producing fewer (planning/control)



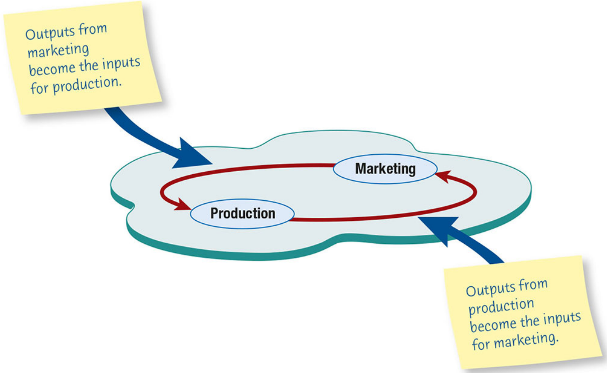
- Ideal System: Self-correcting/self-regulating
  - Inventory system automatically adjusts production/dyeing based on real-time sales data (Italian knitwear co.)
- Environment: Everything outside the boundary. Crucial influences:
  - Community (demographics, location)
  - Economic (markets, competition)
  - Political (government regulations)
  - Legal (laws, guidelines)

Open vs. Closed and The Rise of Virtual

- Openness/Closedness: Exists on a continuum internally too
  - Refers to flow of information/interaction *within* the organization
  - No system is completely open or closed
- Virtual Organizations/Teams
  - Not tied to a single physical location
  - Use technology (networks, communication tools) to connect people
  - Can adapt quickly to changing needs
  - Benefits: Reduced facility costs, rapid response, work-life balance [1]
  - Challenges: Maintaining culture, social connection (e.g, the need for university swag!)
  - Many System Analysis/Design teams work virtually

The Power of Perspective

- Systems Perspective: Allows analysts to understand the *whole* business
- Crucial Insight: Subsystems *rely* on each other. Their work is interrelated
- The Danger: Managers often see the organization through the lens of their *own* department


Outputs from marketing become the inputs for production.
Marketing
Production
Outputs from production become the inputs for marketing.

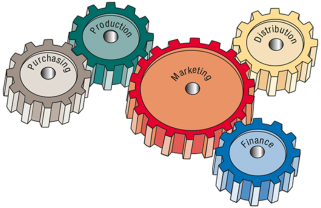
@2024 Pearson Education, Inc.

Warped Views: Managerial Perspectives

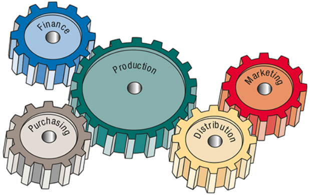
- The Problem: Managers may overemphasize their own function’s importance

- Implication: If these managers rise to strategic levels, their biased view can lead to poor decisions for the overall organization

It’s six.
It’s nine.



How a Marketing Manager May View the Organization



How a Production Manager May See the Organization

Integrating the Organization: ERP Systems

- Concept: Enterprise Resource Planning (ERP) systems aim to integrate information flow across function areas
    - Example: SAP, Oracle Cloud
- Goal: Provide a unified view of the organization's data and processes
- Approach: Usually purchased software, then customized
- Key Difference: Often requires changing business processes to fit the software (unlike traditional analysis which designs processes first)
- Analyst Role: Managing interfaces between ERP and existing ('legacy') systems

ERP: The Game Changer

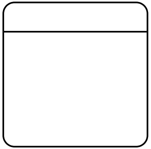
- Complexity: ERP implementations are massive projects impacting *everything*
- Key Hurdles
  - User acceptance (getting people to use it correctly)
  - Integrating with legacy systems and supply chain partners
  - Upgrading complex modules
  - Reorganizing work and decision-making
  - Expanded reach (connecting multiple organizations)
- Impact: Changes job design, required skills, and even company strategy
- Long-term View: Can make employees more effective, but requires careful planning, especially regarding user experience

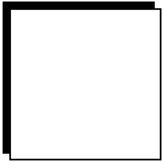
Visualizing the System: Graphical Models

- Why: To understand and communicate:
  - System boundaries (what's *in*, what's *out*)
  - Information flow and usage
  - Relationships between system components
- Think: Blueprints and Maps for the System
- Two Key Early Models
  - Context-Level Data Flow Diagram (DFD)
  - Entity-Relationship (E-R) Model

Zooming Out: The Context-Level DFD

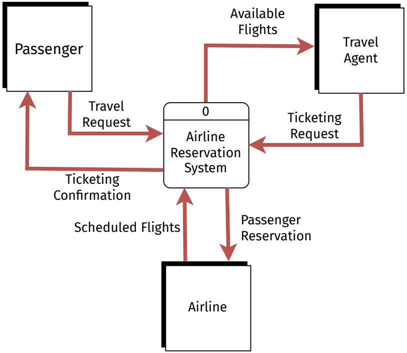
- Purpose: Shows the system's *context* – how it interacts with the outside world
- Think: A higher-level "bird's-eye view"
- Focus: Data flowing IN and OUT, and the overall system process
- Key Components: Only 3 Symbols!
  - Process: The entire system (rounded rectangle). *Transforms data.*
  - External Entity: Sends/Receives data (shaded square). *Outside the system.*
  - Data Flow: Movement of data (arrow)

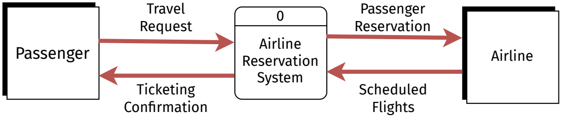




Context-Level DFD Example: Airline Reservations

- Shows the system boundary and interactions clearly
- Example
  - Before Online Booking: Passenger and Travel Agent interact with the System; System interacts with Airline
  - After Online Booking: Travel Agent entity removed; Passenger interacts directly
- Illustrates how context changes affect the diagram





Context-Level DFD: Scope and What It Doesn’t Show

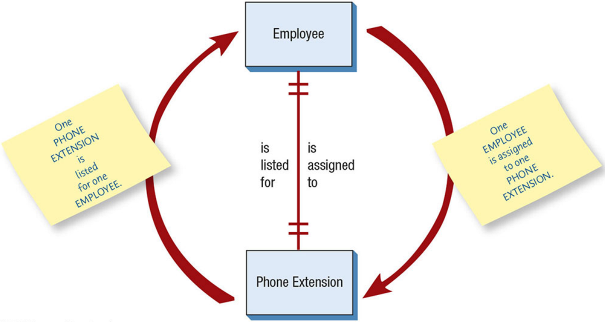
- Strengths
  - Excellent for defining **scope** (what’s included vs. excluded)
  - Shows key external interactions
  - Easy to understand and create
- Limitations
  - Shows *no* internal details (the “how”)
  - Doesn’t show data storage
  - Data flows are high-level (e.g., “Passenger Reservation” - details hidden)
- Use: The very first step in understanding system boundaries. A prerequisite for more detailed diagrams

Modeling Data Structures: Entity-Relationship (E-R) Models

- Alternative View: Focuses on the *data* the system needs, not the *processes*
- Core Components
  - Entities (Rectangles): People, places, objects, events, or concepts about which data is stored (e.g., `CUSTOMER`, `PRODUCT`, `ORDER`)
  - Relationships (Lines connecting entities): How entities are associated or interact (e.g., a `CUSTOMER` *places* an `ORDER`)
- Purpose
  - Defines the structure of the data required by the organization/system
  - Foundation of database design

E-R Basics: Reading Relationships (1:1)

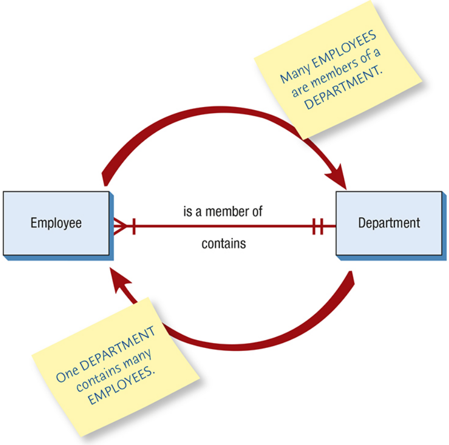
- Cardinality: Specifies *how many* instances of one entity relate to *how many* instances of another
- Example: One-to-One (1:1)
  - One EMPLOYEE is assigned to exactly one PHONE EXTENSION
  - One PHONE EXTENSION is listed for exactly one EMPLOYEE
  - | | (two short parallel marks) often means 'exactly one'


Employee
Phone Extension
is listed for
is assigned to
One PHONE EXTENSION is listed for one EMPLOYEE.
One EMPLOYEE is assigned to one PHONE EXTENSION.

@2024 Pearson Education, Inc.

E-R Relationships: One-to-Many (1:M)

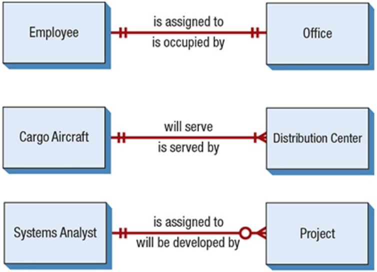
- Example: One-to-Many (1:M)
  - Many EMPLOYEEs *are members of* one DEPARTMENT (Many side)
  - One DEPARTMENT *contains* many EMPLOYEEs (One side)
- Notation
  - $>|$ (Crow's foot) often means 'many' (zero, one, or more)
  - $||$ still means 'exactly one'
- Read the relationship name *towards* the entity you're going to

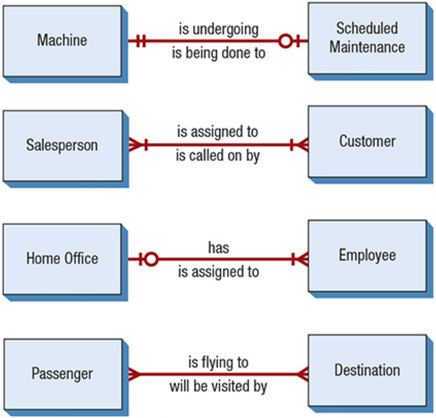

Many EMPLOYEES are members of a DEPARTMENT.
Employee
is a member of
contains
Department
One DEPARTMENT contains many EMPLOYEES.

@2024 Pearson Education, Inc.

E-R Notation: Cardinality and Optionality

- Common Crow's Foot Symbols
  - | |: Exactly One
  - > |: One or More (Many)
  - ⊙|: Zero or One (Optional)
  - > 0: Zero, One, or More (Optional Many)

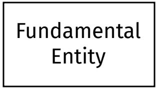




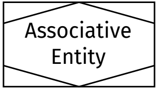
@2024 Pearson Education, Inc.

Types of Entities

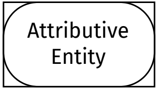
- Fundamental Entity (Standard Rectangle)
  - Represents a basic object or concept (person, place, thing)
  - Example: STUDENT, DEPARTMENT, COURSE
- Associative Entity (Rectangle with diamond inside corners)
  - Links two or more other entities, often representing an event or transaction that depends on them
  - Used especially for M:M relationships
  - Example: An REGISTRATION links a COURSE and a STUDENT.
- Attributive Entity (Rectangle with curved corners)
  - Describes attributes, especially for repeating groups, of another entity
  - Dependent on a fundamental entity
  - Example: OFFERING might be an attributive entity holding specific semester and/or year for a COURSE



Fundamental Entity



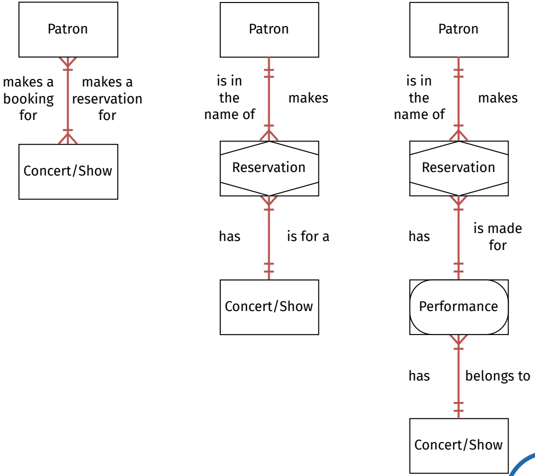
Associative Entity



Attributive Entity

E-R Example: Concert Reservations

- Evolution of Understanding
  - Initial: Too simple?
  - Adding Associative: Better, Reservation links them
  - Adding Attributive: Handles multiple performances
- Shows how E-R models evolve as understanding deepens
- Attributes listed for each entity. Underlined = Key/Searchable



Why Use E-R Diagrams Early?

- Critical Analyst Task: Start drawing E-R diagrams *early* in the project
- Benefits
  - Defines the **scope** and key entities of the problem
  - Helps understand the **business** itself (What are the key things they deal with?)
  - Helps ensure the **right problem** is being addressed
  - Provides a foundation for **data gathering** (interviews, observations, etc.)
  - Needs to be **confirmed** and **revised** as you learn more
- Don't wait for database design! It's an analysis tool first.

A Different Perspective: Use Case Modeling

- What: A way to describe *what* a system does from the *user*'s perspective, without detailing *how*
- Focus: System requirements and user interactions
- Origin: Unified Modeling Language (UML), but now widely used (SDLC, Agile)
- Think: Describing user goals and how the system helps achieve them

Use Case Basics: Actors and Use Cases

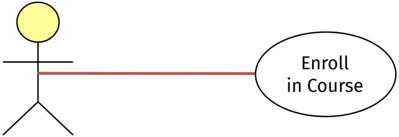
- Actor: Represents a role played by a user (human, another system, device) interacting with the system
  - Exists *outside* the system boundary
  - Can initiate or participate in a use case
  - Same person can be multiple actors (e.g., Employee, Customer)
  - Symbol: Typically a stick figure
  - Primary Actors: Initiate use cases, main users (e.g., Student)
  - Supporting Actors: Help maintain the system (e.g., Help Desk Staff)
- Use Case: Represents a specific goal or task an actor wants to achieve with the system
  - Describes a sequence of interactions
  - Should provide something of *value* to the actor
  - Named with Strong Verb-Noun (e.g., “Enroll in Course”, “Submit Order”)
  - Symbol: Typically an oval

How They Connect: Use Case Relationships

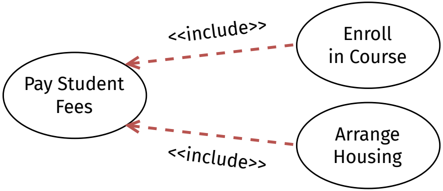
- Behavioral Relationships: Show how actors and use cases interact or relate to each other
- Four Main Types

Relationship	Symbol	Purpose
Communicates	———	Actor and Use Case interaction
Includes	<<include>> ←- - - - - - - - - -	One use case uses *another* common use case
Extends	<<extend>> - - - - - - - - - - -→	One use case handles exceptions/variations for another
Generalizes	———▷	One actor/use case is more general version of another

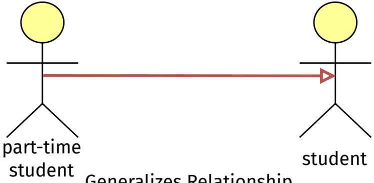
Use Case Relationships Explained



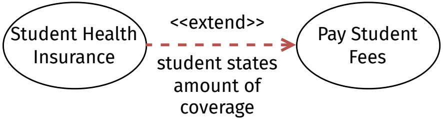
Communicates Relationship



Includes Relationship



Generalizes Relationship



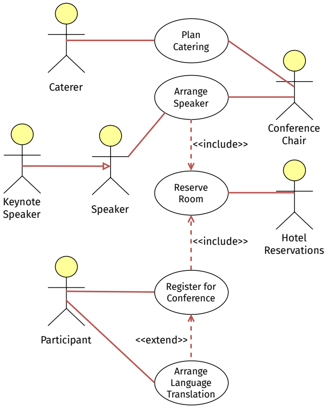
Extends Relationship

Developing Use Case Diagrams

- Purpose: Define system **scope** (actors are outside, use cases inside, communicates lines are boundaries)
- Starting Point
  - User interviews, JAD sessions, Agile stories → Ask “What should the system do for you?”
  - Existing Context-Level DFD
    - External Entities → Potential Actors
    - Data Flows → Potential Use Cases
- Steps
  - Identify **Actors** (review roles)
  - Identify high-level **Events** and **Primary Use Cases** initiated by actors
  - Determine **Variations/Alternative Paths** (these might become «extend» relationships or separate scenarios)
- Goal: Capture significant user interactions and goals. Keep it manageable (20-50 use cases for large systems)

Use Case Diagram Example: Conference Planning

- Scenario: System to help plan an academic conference
- Actors
  - Conference Chair
  - Participant
  - Speaker
  - Keynote Speaker
  - Hotel Reservations (Systems?)
  - Caterer



Beyond the Diagram: Use Case Scenarios

- Need More Detail: Diagrams show *what* happens, scenarios describe *how* (step-by-step)
- Use Case Scenario: A textual description detailing the sequence of steps within a specific use case
  - Primary Path (Standard Flow/“Happy Path”): Normal successful completion
  - Alternative Paths: Variations, error conditions, exceptions
- Purpose: Elaborate on the use case oval, providing necessary detail for design and development
- Format: Often uses a standardized organizational template

Use Case Levels (Cockburn’s Altitude Metaphor)

- Different Levels of Detail: Not all use cases need the same granularity
- Altitude Metaphor [2]
    - White (like clouds): Highest level, enterprise-wide (4-5 total). E.g., “Manage Supply Chain”
    - Kite: High-level summary, business unit/dept. E.g., “Manage Reservations”
    - Blue (Sea Level): User goal level, most common. E.g., “Register Continuing Student” (takes 2-20 mins)
    - Indigo (Fish): Functional/Subfunctional detail. E.g., “Choose a Class”, “Pay Fees”
    - Black (Clam): Most detailed, subfunction/implementation. E.g., “Validate Secure Logon”
- Choose Level Appropriately: Focus often on Blue level for user interaction understanding

Use Case Scenario Structure (Template)

- Common Sections
  - Header
    - Use Case Name and ID
    - System/Application Area
    - Actor(s)
    - Stakeholders (Interested parties, non-actors)
    - Level (Blue, Kite, etc.)
    - Description (Brief goal)
    - Trigger (Event starting the use case - external or temporal)
  - Steps Performed
    - Numbered sequence for the primary path
    - Information required/used for each step
    - Alternative paths/extensions
  - Footer
    - Preconditions (What must be true *before* starting)
    - Postconditions (What is true *after* successful completion)
    - Assumptions (About environment, user, tech)
    - Minimal Guarantee (Worst acceptable outcome)
    - Success Guarantee (Desired outcome)
    - Outstanding Issues/Questions
    - Optional: Priority, Risk, Requirements Met

Demo: Use Case Scenario - Conference Registration

Use Case Scenario Example Details

- Example (Conference Registration)
  - Shows detailed steps for the main path
  - Includes conditional logic within steps (e.g., “IF multiple airports…”)
  - Can reference alternative scenarios/extensions (Flight selection, Seat selection)
  - May include looping/iterative steps
- Key: Captures the flow and necessary logic clearly for developers

Demo: Use Case Scenario - Airline Reservation

Why Use Cases are Helpful

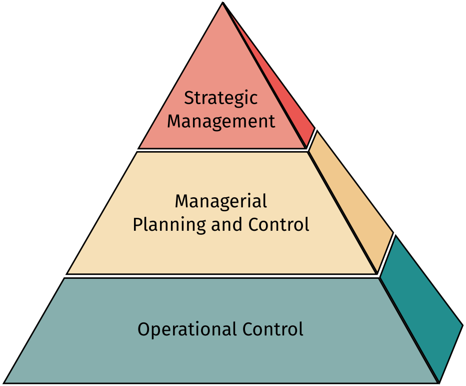
- Communication: Simple diagrams and stories are easy for non-technical users and stakeholders to understand
- Requirements Focus: Keeps focus on *what* the system needs to do for the user
- Boundary Definition: Clearly shows system scope and interactions
- Traceability: Links user needs to design elements and test cases
- Versatility: Useful across different development methodologies (SDLC, Agile, O-O)
- Foundation: Scenarios provide detailed steps for process modeling and test planning

Changing Perspectives: Levels of Management

- Organizations aren't flat: Management exists at different levels
- Why does this matter for Systems Analysis? → Different levels have different
  - Responsibilities and Goals
  - Decision-making styles
  - Information Needs
- Understanding these levels helps us design systems that serve *everyone* effectively
- Think: Different altitudes give different views of the landscape

The Three Tiers of Management

- Typical Structure: A three-tiered hierarchy
  - Top: Looking Outward and Forward
  - Middle: Bridging Strategy and Operations
  - Bottom: Managing Daily Activities
- Think: The Captain, Officers, and Crew of the ship



Level 1: Operational Control

- Focus: Direct supervision of daily operations and transactions
- Decisions
  - Based on predetermined rules
  - Predictable outcomes
  - Short-term horizon
  - Examples: Work scheduling, inventory control, shipping/receiving process control
- Problem Type: Structured, well-defined problems. Easy to identify
- Alternatives: Usually clear and easy to list
- Decision Frequency: Repetitive

Level 2: Middle Management

- Focus
  - Short-term planning and control
  - Allocating resources to meet objectives
- Decisions
  - How to best use resources (people, budget, equipment)
  - Monitoring performance against plans/standards
  - More variety than operational level
  - Examples: Developing departmental budgets, deciding on staffing levels, scheduling production for the next quarter
- Problem Type: Mix of structured and semi-structured problems
- Alternatives: May require more analysis than operational level
- Decision Frequency: Less repetitive than operational

Level 3: Strategic Management

- Focus: Long-term direction, looking outward at environment, setting overall goals
- Decisions
  - Define organizational mission and goals
  - Shape the organization's future (years ahead)
  - High uncertainty, broad scope
  - Examples: Entering new markets, developing new product lines, acquiring other companies, setting corporate policy
- Problem Type: Unstructured or semi-structured, often ambiguous. Difficult to identify.
- Alternatives: Often unclear, difficult to articulate or evaluate.
- Decision Frequency: Often unique, one-time decisions

Information Needs: Operational Control

- Primary Need: Internal, current performance data
  - Highly dependent on **real-time** or near real-time information
  - Repetitive, low-level detail
- Moderate Need
  - Past performance (for immediate comparison)
  - Period reports (daily/weekly summaries)
- Example Systems: Transaction Processing Systems (TPS), real-time dashboards showing production status, inventory levels

Information Needs: Middle Management

- Primary Need: Mix of internal information - current and historical
  - High need for **real-time** info (troubleshooting, control)
  - High need for **current performance** vs. **standards**
  - High need for **historical information** (trend analysis)
  - Need for **predictive information** and “what-if” scenarios (short-term planning)
- Moderate Need: Some external information (departmental competitors, industry trends)
- Example Systems: Management Information Systems (MIS), Decision Support Systems (DSS) providing summaries, exception reports, budget tracking, forecasting tools

Information Needs: Strategic Management

- Primary Need: External and Internal; Future-oriented
  - High dependence on **external information** (market trends, competitors, economy, regulations)
  - High need for **predictive information** and **"what-if"** scenarios (long-term strategy)
  - High need for **summarized**, **periodic information** (tracking overall progress)
  - Need for **risk assessment** (including security)
- Moderate Need: Summarized internal performance (the big picture)
- Low Need: Detailed, real-time internal operational data
- Example Systems: Executive Information Systems (EIS), DSS, Business Intelligence (BI) tools providing high-level dashboards, market analysis, long-range forecasting, risk modeling

Implications for IS Development

- No Single System Fits All: Information systems must be tailored to the target management level
- Operational Level Needs: Systems emphasizing transaction processing, accuracy, real-time monitoring, detailed reporting
- Middle Management Needs: Systems providing summaries, exception reporting, trend analysis, short/medium-term forecasting, DSS capabilities
- Strategic Level Needs: Systems integrating external data, supporting long-range planning, complex scenario modeling, high-level summarization (EIS/BI)
- Analyst’s Job: Understand the user’s level and specific decision-making context to determine appropriate system features and information presentation

Collaborative Design Across Levels

- Concept: Involving stakeholders from different levels (operational, middle, strategic) and departments in the system design process [3], [4]
- Goal: Create systems that meet diverse needs and gain buy-in
- Challenges
  - Power dynamics based on hierarchy
  - Differing perspectives and priorities
  - Ensuring relevant information flows to all participants
- Opportunity: Leveraging expertise from all levels (e.g., operational users know workflows, middle managers know constraints, strategic users know goals)
- Key
  - Structure collaboration carefully
  - Value diverse expertise
  - Manage information flow

Beyond Structure: Organizational Culture

- What: The shared values, beliefs, attitudes, and norms that shape behavior within an organization
  - The organization's "personality"
  - Often unwritten, but powerful
- Why Care: Culture significantly influences
  - How people work together
  - How decisions are *really* made
  - How readily new ideas (and systems!) are accepted
- Think: The invisible forces guiding how things get done

Not Just One Culture: Subcultures

- Reality: Organizations rarely have a single, uniform culture
- Subcultures: Smaller groups within the organization with their own distinct norms, values, and ways of communicating
    - Can form around departments, teams, roles, projects, even virtual teams
    - May align with, or sometimes conflict with, the “official” culture
    - Can compete for influence (“Our way is better!”)
- Challenge: Identifying and understanding these often overlapping and dynamic subcultures

Reading the Signs: Cultural Symbolism

- How to Perceive Culture/Subcultures? Look for symbols
  - Verbal Symbolism
    - Shared language, jargon, acronyms
    - Myths and stories about company history/heroes
    - Metaphors used (“We’re a family,” “We’re a war machine”)
    - Humor and inside jokes
  - Nonverbal Symbolism
    - Artifacts (awards, posters, logos)
    - Rites and ceremonies (parties, award events, retirement rituals)
    - Dress code (formal vs. casual)
    - Office layout and decoration (open plan vs. private offices, fancy vs. functional)

Culture’s Influence on Information Systems

- Major Impact: Subcultures strongly influence IS requirements, adoption, and use
- Potential Issues
  - Resistance to Change: If a new system clashes with a subculture’s values, workflow, or status
  - Information Hoarding: Subcultures might restrict information flow perceived as empowering rivals
  - System Acceptance: May be embraced by one group but rejected by another
  - Determining Requirements: Different subcultures may have conflicting needs or priorities for a system
- Analyst Goal: Understand subcultures to anticipate challenges and tailor implementation (e.g., targeted training)

Technology Shaping Culture: The Slack Example

- Technology isn't neutral: Tools can influence and shape organizational culture
- Example: Slack (Workplace Messaging App)
  - Less Formal: Than email, encourages quicker, more casual communication [5]
  - Channel Structure: Public (archived, searchable by team), Private (invitation-only), Direct Messages
  - Potential Impacts
    - Fosters collaboration and quicker communication
    - Can flatten hierarchy (easier access across levels?)
    - May create its own etiquette and norms (emoji use, response times)
    - Can potentially blur work/life boundaries
    - Shapes *how* colleagues interact and build relationships
- Other Tools: Similar impacts from collaborative platforms (Teams, Discord, etc.), project management software, video conferencing

Leveraging Culture Understanding

- Be Observant: Pay attention to verbal and nonverbal cues
- Listen Actively: Understand the stories, concerns, and perspectives of different groups
- Identify Influencers: Who are the key people respected within different subcultures?
- Anticipate Resistance: Where might clashes occur between the system and existing culture?
- Tailor Approach
  - Communication: Frame benefits relevant to specific subcultures
  - Training: Address specific concerns and workflows
  - Implementation: Consider pilot groups or phased rollouts based on cultural readiness
- Goal: Work *with* the culture(s) where possible, mitigate conflicts, and facilitate smoother system adoption

References I

[1] T. Henderson, “In-Person Workers Are Slow to Return to Jobs, Data Shows,” Published by Stateline, a project of The Pew Charitable Trusts. (2021), [Online]. Available: https://www.pewtrusts.org/en/research-and-analysis/blogs/stateline/2021/09/09/in-person-workers-are-slow-to-return-to-jobs-data-shows (cit. on pp. 16–19).

[2] A. Cockburn, *Writing Effective Use Cases*, 1st. Boston, MA: Addison-Wesley Professional, 2000 (cit. on pp. 93–99).

[3] N. Levina, “Collaborating on Multiparty Information Systems Development Projects: A Collective Reflection-in-Action View,” *Information Systems Research*, vol. 16, no. 2, pp. 109–130, 2005. DOI: 10.1287/isre.1050.0055. [Online]. Available: https://pubsonline.informs.org/doi/abs/10.1287/isre.1050.0055 (cit. on pp. 140–143).

[4] A. F. Phelps, *The Collective Potential: A Holistic Approach to Managing Information Flow in Collaborative Design and Construction Environments*, 1st. North Bend, WA: Turning Point Press, 2012 (cit. on pp. 140–143).

References II

[5] Akumina, “2022 State of the Digital Workplace & Modern Intranet Report,” Akumina, Nashua, NH, Tech. Rep., 2022, White paper. [Online]. Available: https://www.akumina.com/resources/2022-state-of-digital-workplace-report (cit. on pp. 156–159).