PENGEMBANGAN SPK

Siklus hidup – SDLC tradisional

Prototyping metodologi pengembangan

Manajemen perubahan

Perangkat dan tingkat teknologi SPK

Platform pengembangan SPK

Memilih perangkat pengembangan SPK

SPK yang dikembangkan oleh tim atau individu

Mengembangkan SPK bersama

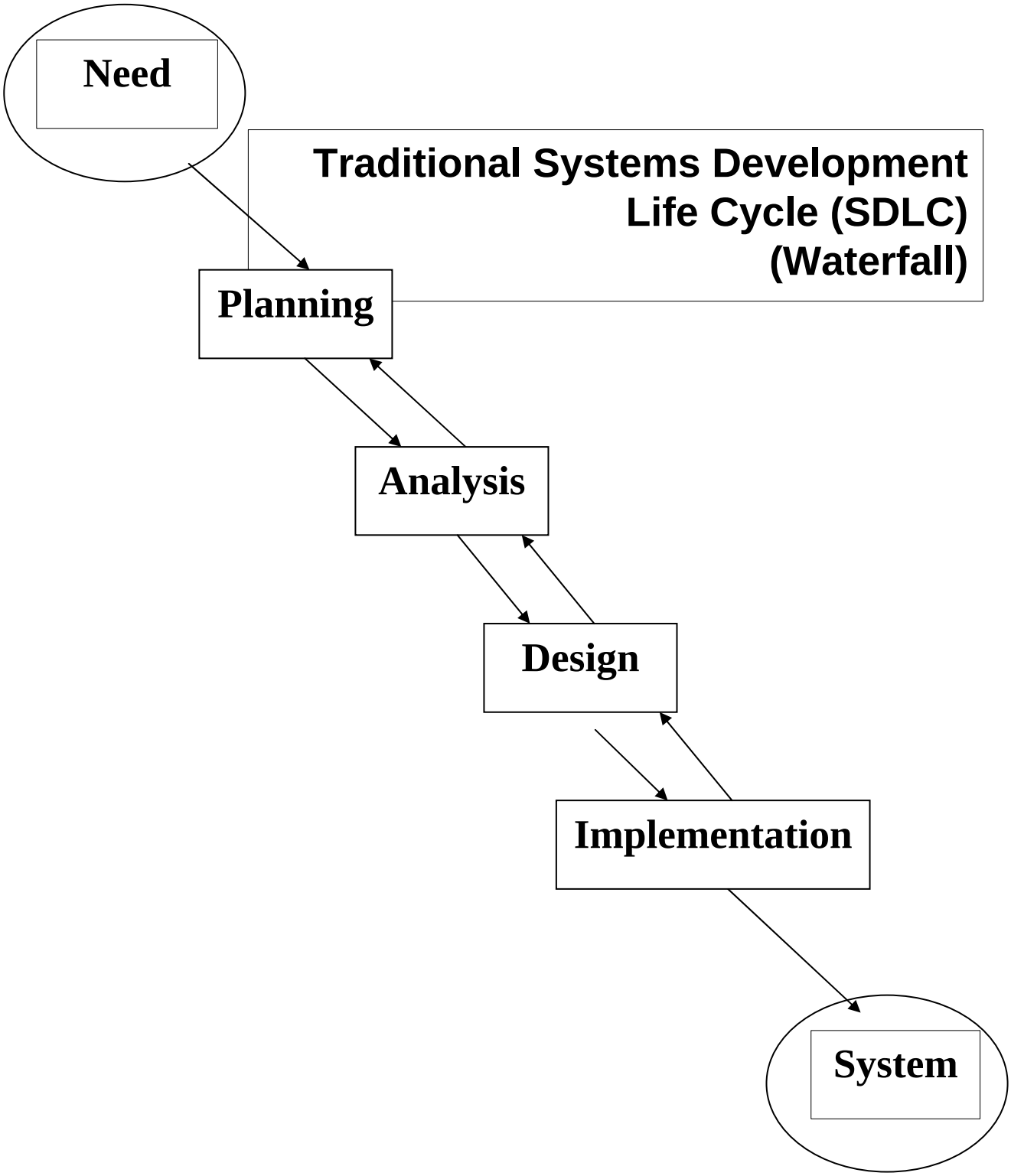
Referensi lihat SAP : [5] Bab 6,
[7] Chapter 6, [8] Marakas-14

System Development Issues

- System development life cycle (SDLC)
- Prototyping
- Forming the development team
- Complex process
- Technical issues
- Behavioral issues
- Different approaches

Fundamental SDLC Phases

- Planning
- Analysis
- Design
- Implementation


Need
Traditional Systems Development Life Cycle (SDLC) (Waterfall)
Planning
Analysis
Design
Implementation
System

Planning Why Build the System?

Minor Step	Deliverable
1. Identify business value	System request
2. Analyze feasibility	Feasibility study
3. Develop work plan	Work plan
4. Staff project	Staffing plan, Project charter
5. Control and direct project	Project manag. tools CASE tool Standards list Project binders / files Risk assessment

Analysis Who, What, When, Where?

Minor Step	Deliverable
6. Analyze problem	Analysis plan
7. Gather information	Information
8. Model process(es)	Process model
9. Model data	Data model

Design How Will the System Work?

Minor Step	Deliverable
10. Design physical system	Design plan
11. Design architecture	Architecture design, Infrastructure design
12. Design interface	Interface design
13. Design database and files	Data storage design
14. Design program(s)	Program design

Implementation System Delivery

Minor Step	Deliverable
15. Construction	Test plan, Programs, Documentation
16. Installation	Conversion plan, Training plan

Common Implementation Headaches (DSS in Focus 6.4)

- No project team or management support
- Hazy purpose; no defined schedule; ballooning scope
- Unclear aspects of make vs. buy decisions
- Few project integrations are functional out of the box
- Qualitative benefits
- No user buy in
- Poor project management skills
- No accountability / no responsibility

CASE Tools

- Information systems for systems analysts
- Can help manage system development
- Upper CASE (assists in analysis)
- Lower CASE (manages diagrams and code generation)
- Integrated CASE (both)
- Oracle Enterprise Development Suite
- Rational Rose
- Paradigm Plus
- Visible Analyst
- Logic Works Suite
- AxiomSys and AxiomDsn
- V32 & X32
- Visual Studio

Project Management (PM)

- Team leader must have good PM skills
- Major reason for IS development failures-bad PM skills
- Only 26% of all projects surveyed (23,000) in 1998 succeeded
- 28% failed, 46% challenged
- Lower success rates for large companies
- Better PM skills needed

Skills for Project Managers

- Technology and business knowledge
- Judgment
- Negotiation
- Good communication
- Organization

Alternative Development Methodologies

- Parallel development
- Rapid application development (RAD) methodologies
  - Phased development
  - Prototyping
  - Throwaway prototyping

Parallel Development

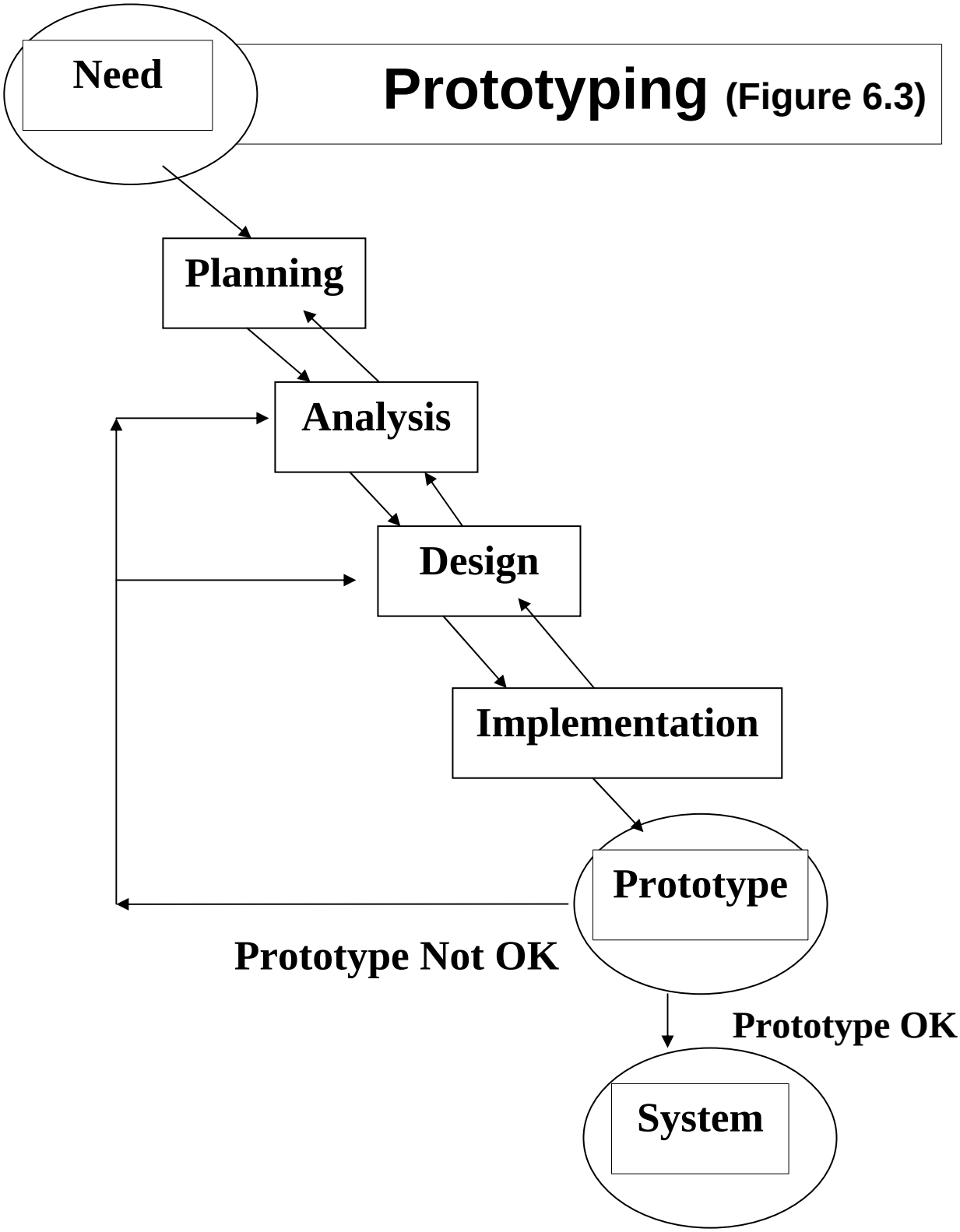
- Multiple copies of design and implementation phases
- To develop separate subsystems
- All come together in a single implementation phase

Phased Development

- Break system up into versions developed sequentially
- Each version has more functionality
- Evolves into a final system
- Users gain functionality quickly
- But initial systems are incomplete

Prototyping

- Performing analysis, design, and implementation phases concurrently, and repeatedly
- Users see system functionality quickly and provide feedback
- Decision maker learns about problem
- But can lose gains in repetition


Need
Prototyping (Figure 6.3)
Planning
Analysis
Design
Implementation
Prototype
Prototype Not OK
Prototype OK
System

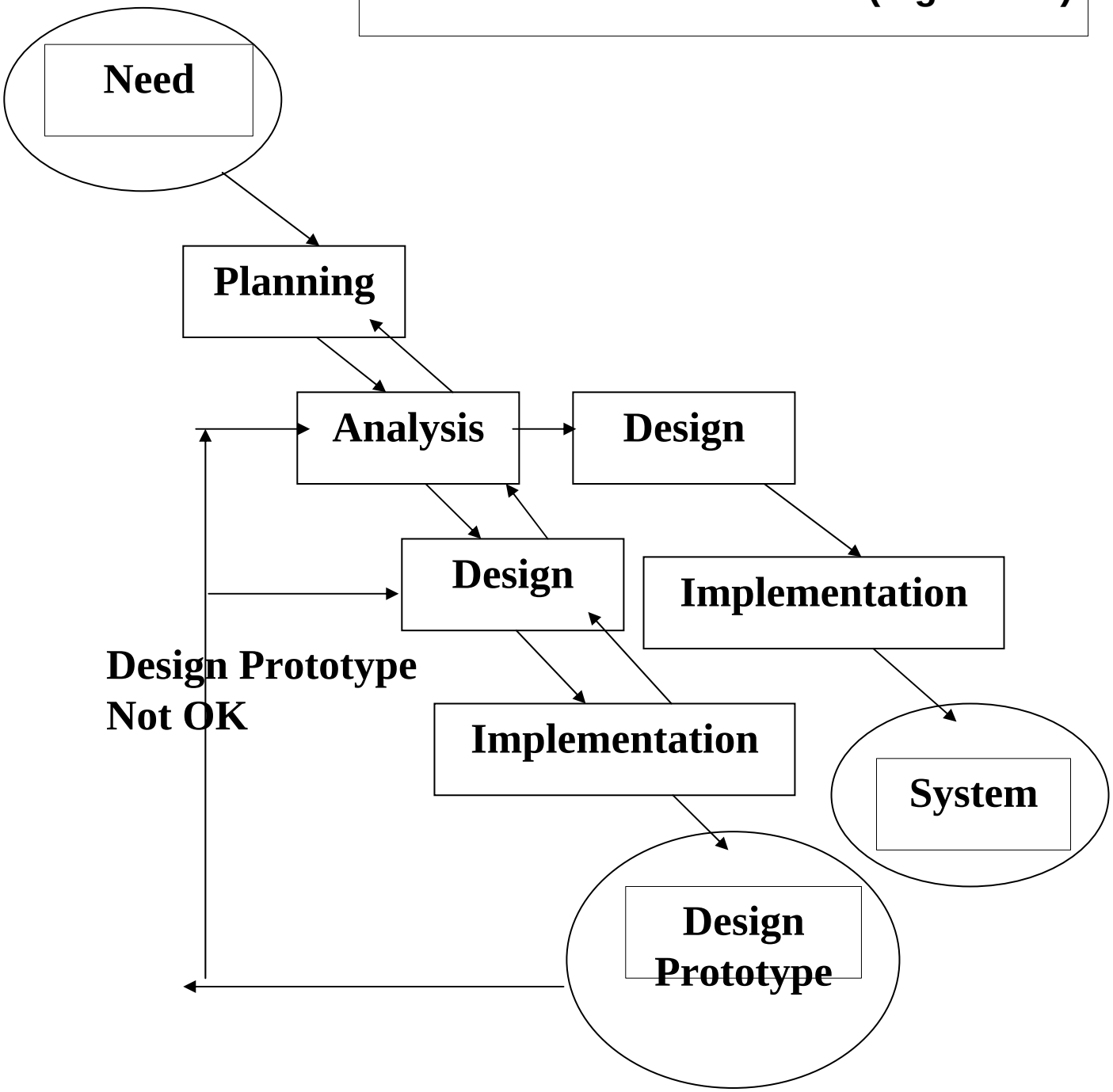
Throwaway Prototyping

- Like prototyping and SDLC
- Analysis phase is thorough
- Design prototypes assist in understanding the system
- Example: can use Excel, then Visual Basic
- (Figure 6.4)

Prototyping for DSS Development

- Problems are semistructured or unstructured
- Managers and developers may not completely understand problem
- Use prototyping

Throwaway Prototyping
(Figure 6.4)



Prototyping Terms

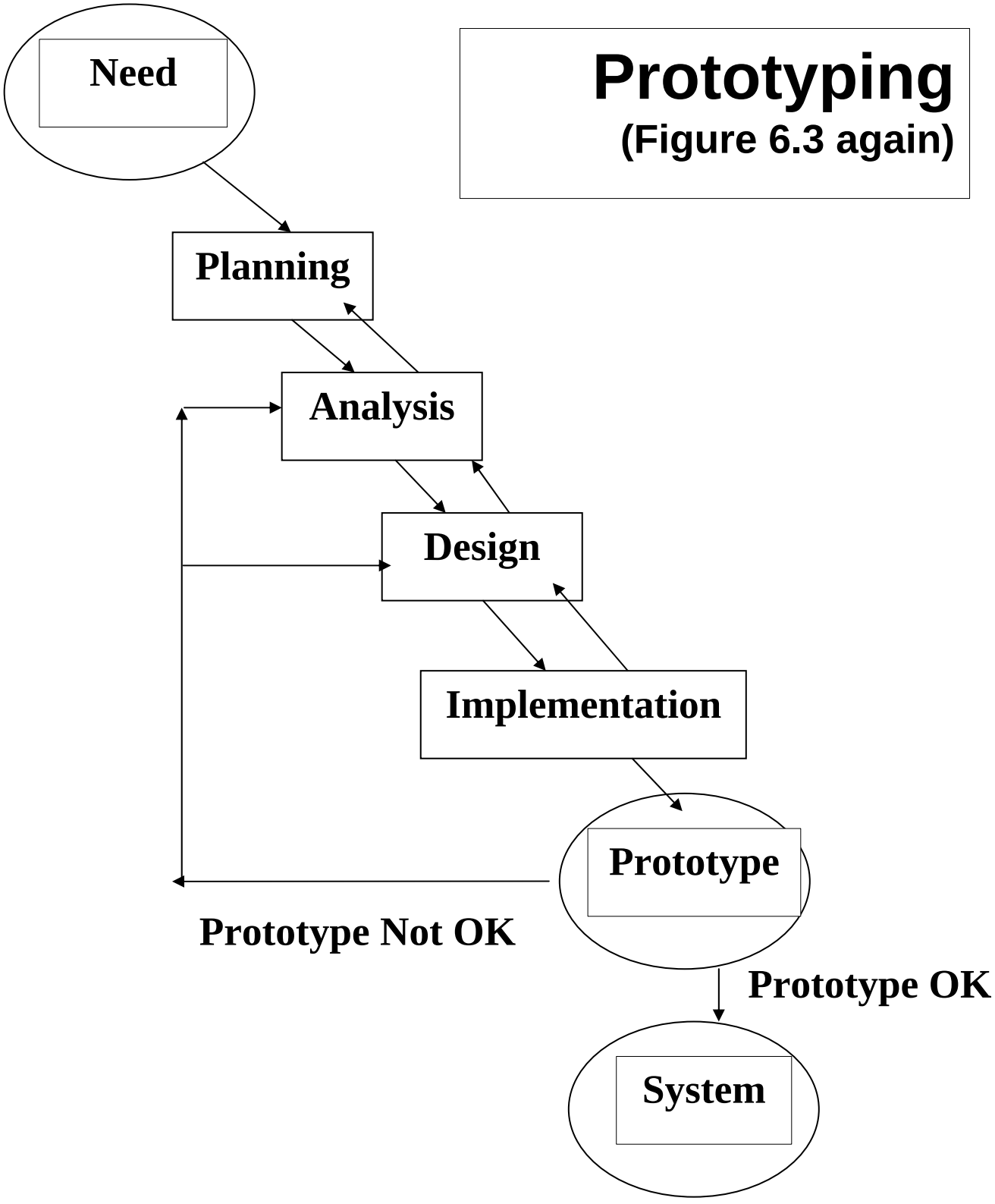
- Iterative design
- Evolutionary development
- Middle-out process
- Adaptive design
- Incremental design

Prototyping Examples

- Opening Vignette: InfoNet HR Portal System at Osram Sylvania

- Case Application 6.1: POP DSS at IMERSY

Prototyping
(Figure 6.3 again)



Why Prototyping?

- Users and managers involved in every phase and iteration
- Learning is part of design
- Prototyping bypasses the information requirement definition (step 7)
- Short interval between iterations
- Initial prototype must be low cost

Advantages

- Short development time
- Short user reaction time
- Improved user understanding
- Low cost

Disadvantages

Gains may be lost in

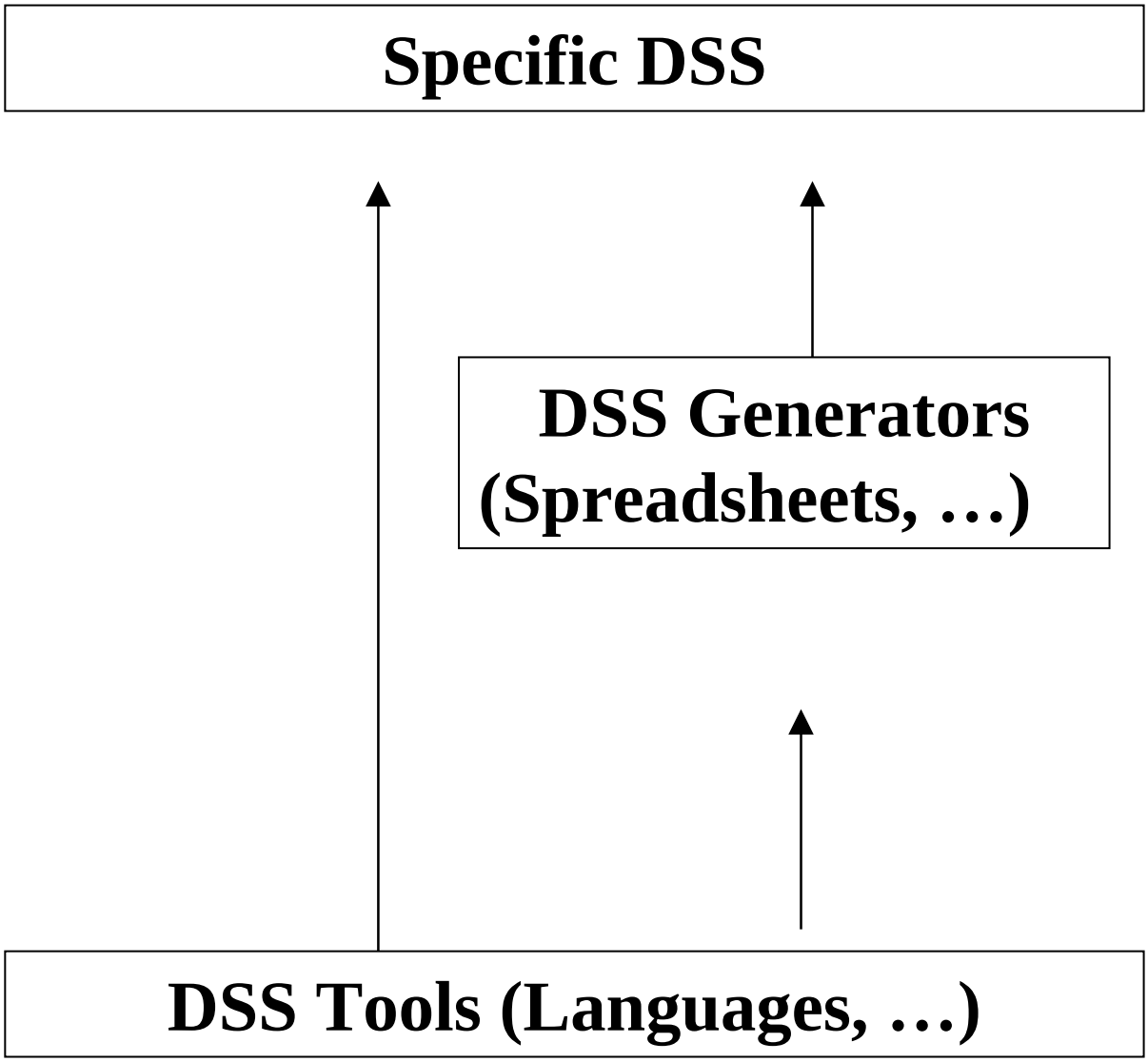
- Thorough understanding IS's benefits and costs
- Detailed description of information needs
- Easy to maintain IS design
- Well-tested IS
- Well-prepared users

DSS Technology Levels and Tools

- Three Levels of DSS Technology
  - Specific DSS [the application]
  - DSS integrated tools (generators) [Excel]
  - DSS primary tools [programming languages]
- Plus
  - DSS integrated tools

- Now all with Web hooks and easy GUI interfaces
- Relationships among the three levels (Figure 6.5)

DSS Technology Levels
(Figure 6.5)



DSS Development Platforms

- General-purpose programming language
- Fourth-generation language (4GL)
- OLAP with a data warehouse or large database
- DSS integrated development tool (generator, engine)
- Domain-specific DSS generator
- Use the CASE methodology
- Integrate several of the above

Hardware Selection

- PCs
- Unix workstations
- Network of Unix workstations
- Web servers
- Mainframes

- Typically use existing hardware

Software Selection

Complex because

- At start, information requirements, etc. are unknown
- Hundreds of packages
- Software updated rapidly
- Price changes
- Many people involved in decision
- Language capability problems
- Different tools might be needed
- Many criteria
- Technical, functional, end-user, and managerial issues
- Inaccurate published software reviews
- Might prefer a single vendor

- Maybe use the AHP!!!

Team-Developed DSS

- Substantial effort
- Extensive planning and organization
- Some generic activities

- Group of people to build and to manage it
  Size depends on
  - Effort
  - Tools

Team-Developed Versus User-Developed DSS

- DSS 1970s and early 1980s

- Large-scale, complex systems
- Primarily provided organizational support
- Team efforts

End-User-Developed Systems

- Personal computers
- Computer communication networks
- PC-mainframe communication
- Friendly development software
- Reduced cost of software and hardware
- Increased capabilities of personal computers
- Enterprise-wide computing
- Easy accessibility to data and models
- Client/server architecture
- Now OLAP

Balance

Organizational Placement of the DSS Development Group

1. Information services (IS) department
2. Highly placed executive staff group
3. Finance or other functional area
4. Industrial engineering department
5. Management science group
6. Information center group

End-user Computing and User-Developed DSS

- End-user Computing (end-user development): development and use of computer-based information systems by people outside the formal information systems areas
- End-users
  - At any level of the organization
  - In any functional area
  - Levels of computer skill vary
  - Growing

User-Developed DSS Advantages

1. Short delivery time
2. Eliminate extensive and formal user requirements specifications
3. Reduce some DSS implementation problems
4. Low cost

User-Developed DSS Risks

1. Poor Quality
2. Quality Risks
   - Substandard or inappropriate tools and facilities
   - Development process risks
   - Data management risks
3. Increased Security Risks
4. Problems from Lack of Documentation and Maintenance Procedures

Issues in Reducing End-User Computing Risks

- Error detection
- Use of auditing techniques
- Determine the proper amount of controls
- Investigate the reasons for the errors
- Solutions
- Spreadsheet errors
  - Should use same controls as normal IS

Developing DSS: Putting the System Together

- Development tools and generators
- Use of highly automated tools
- Use of prefabricated pieces

- Both increase the developer's productivity

DSS Development System Includes

- Request (query) handler
- System analysis and design facility
- Dialog management system
- Report generator
- Graphics generator
- Source code manager
- Model base management system
- Knowledge-base (management) system
- Object-oriented tools
- Standard statistical and management science tools
- Special modeling tools
- Programming languages
- Document imaging tools

DSS Development System Components

- Some may be integrated into a DSS generator
- Others may be added as needed
- Components used to build a new DSS
- Core of system includes development language or DSS generator
- Construction by combining programming modules
- Windows environment handles the interface

DSS Research Directions and The DSS of the Future

- More AI
- Faster, more powerful computers
- The Web - interfaces and DB and model access
- More and better GSS
- ERM/ERP
- Knowledge management
- Better GUI
- Better telecommunications
- More research on theories
- More research on methods

Kesimpulan

- DSS are complex and their development can be too
- SDLC
- Prototyping
- DSS technologies
- DSS teams or individuals
- End user computing
- Tool and generator selection can be tricky
- DSS research continues