Designing and Building
Decision Support Systems

1

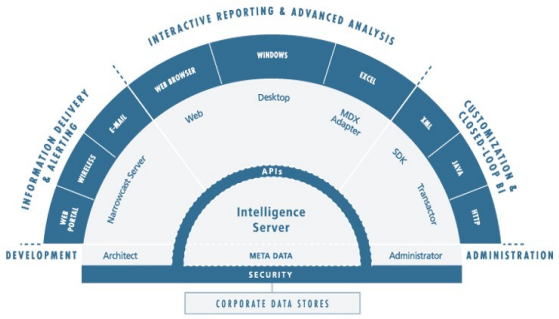
Strategies for DSS Analysis and Design

There are two common strategies for DSS development:

- Programming a customized DSS: either a general purpose language like C++ or a fourth-generation language like Delphi or Visual C11 can be used. This allows for development of special interfaces between the DSS and other applications.
- Employing a DSS generator: these range from spreadsheets such as Excel—perhaps with some add-ins—or a more sophisticated generator such as MicroStrategy's DSS Architect.

2

DSS Architect is Part of MicroStrategy's 7*i* Suite



3

The DSS Analysis and Design Process

Several approaches can be applied to the process of DSS development:

- System development life cycle –employs a series of recursive phases each with its own inputs, activities and outputs. These phases begin with "Problem definition" then "Feasibility Analysis" and finish with "Implementation" and "Maintenance"

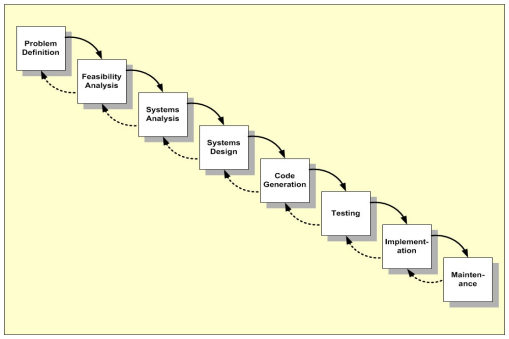
4

The DSS Analysis and Design Process

- The *primary advantage* of SDLC is the structure and discipline it brings. It is often used today, especially in cases where there is a contractual relationship between the DSS developer and the end users.
- The *major complaint* about SDL is its rigidity since requirements in a DSS can change rapidly.

5

Classical System Development Life Cycle



6

The DSS Analysis and Design Process

Besides SDLC, there are two other approaches to DSS development:

- ROMC analysis – this approach asks the developer to understand representations (R), operations (O), memory aids (M), and controls (C). Representations include charts and tables.
- Functional category analysis – the developer identifies the specific functions necessary for a specific DSS from a broad list of available functions.

7

Functional Categories

- Selection – locating knowledge within the knowledge base for use as input
- Aggregation – creation or derivation of summary statistics, such as averages or totals
- Estimation – creation of model parameter estimates
- Simulation – creation of knowledge about expected outcomes or consequences of specific actions

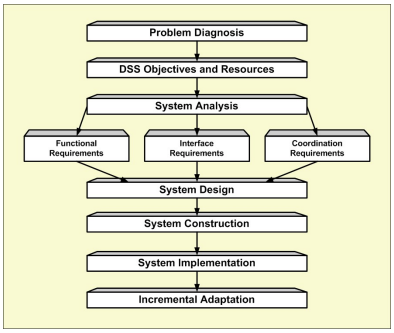
8

Functional Categories (cont.)

- Equalization – creation of knowledge regarding conditions necessary to maintain consistency
- Optimization – discovering what set of parameter values best meet a set of performance measures

9

Generalized DSS Development Process



10

DSS Development Process

For unstructured problems, we employ an alternate development strategy. There are seven basic activities in this process (not all may be performed in every project).

1. Problem diagnosis – formal identification of the problem context
2. Identification of objectives and resources – specific objectives must be described and available resources identified
3. System analysis – three categories of requirements (functional, interface, and coordination) are established.

11

DSS Development Process

The remaining steps are:

4. System design – the determination of components, structure, and platform
5. System construction – an iterative prototyping approach, with small but constant refinement employed
6. System implementation – where testing, evaluation, and deployment occurs
7. Incremental adaptation – this final stage is a continual refinement of the activities of the earlier six stages.

12

SDLC versus DSS Development Process

- SDLC evolved out of developers' experience with computer-based information systems. The sequential and structured nature of the process is one of its primary strengths.
- In practice, a more iterative, bottom-up design approach might work better.
- For DSS development—as opposed to general IS development—problems tend to be less structured and a more evolutionary design approach is needed.

13

Prototyping

- An increasingly popular method of system development. For DSS development, it is usually of an iterative or evolutionary nature.
- Early stages are similar to the classic SDLC methodology until the first prototype is in place. At that point the methods diverge as the prototype undergoes almost constant, small changes.
- This process requires a significantly higher level of interaction between analyst and user.

14

Prototyping versus SDLC

- Throwaway prototypes are used for demo purposes only and then discarded. In DSS development, an iterative prototype is more often used.
- Prototyping often reduces development time and cost over the SDLC approach. Also, the higher level of user involvement can lead to greater support for the DSS from management.
- Advantages to the more cautious approach of SDLC are that documentation is often more comprehensive and there is better understanding of the system's benefits and corresponding costs.

15

The DSS Developer

- At one extreme, the DSS developer is an experienced professional trained in computer science or MIS.
- At the other is a managerial decision maker who perceives a need for computer support.
- Although the novices may be experiencing a development effort for the first time, they possess a more intimate knowledge of what they want the DSS to accomplish. With the right tools, this may give them an advantage.

16

The Necessary Skill Set

- Regardless of experience, the developer needs to possess key skills:
  1. Understanding the problem domain
  2. Understanding specific user requirements
  3. Understanding the available development technologies
  4. Access to appropriate knowledge
- Because all of these skills may not be available in a single person, a team may be required.

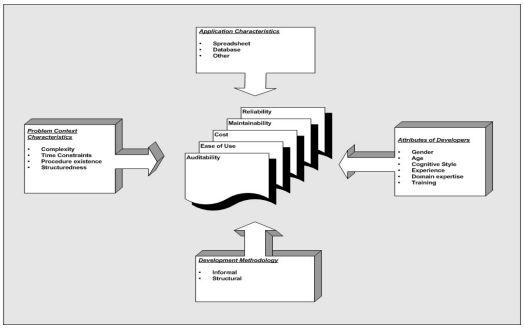
17

End-User DSS Development

- End-user developers are those who fall outside the confines of the IS department.
- End-user developers play a variety of organizational roles and exhibit a variety of computer skills.
- They are as diverse as "just a guy with a problem to solve" to the "department computer guru".
- Most end-user-developed applications evolve from an informal process, which may cause problems if the application needs to be integrated into a larger DSS.

18

Factors Influencing Risks and Outcome Characteristics of End-User-Developed Applications



19

Advantages and Risks to End-User Development

- Assuming the end user has the required skills and tools, a major advantage is reduction of delivery time.
- Others are reduced time in gathering end-user specifications and fewer implementation problems.
- All these lead to lower cost of development as well as faster implementation.

20

Advantages and Risks to End-User Development

- One disadvantage is that novice developers may bypass conventional control and testing procedures.
- Another is lack of quality documentation, which can be a major problem if the developer leaves the organization.
- Lack of security measures also tend to be a problem, especially on applications that access the Internet.

21

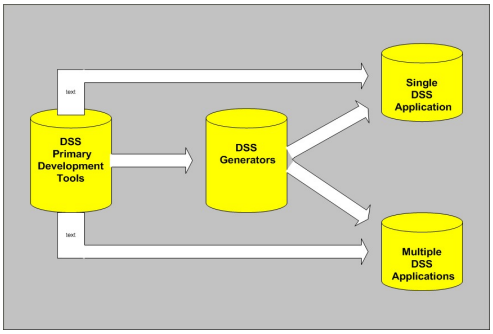
Tools for DSS Development

There are a variety of tools available, roughly falling into three categories:

1. Primary development tools – these include programming languages and database query mechanisms.
2. DSS generators – at a higher level of technology, these possess integrated, diverse functionality, including decision modeling, sophisticated reporting, and database management.
3. Specific DSS applications – for some problem types there may be a commercially available package that can be acquired and customized.

22

DSS Development Tool Classification



23

Development Tool Selection Criteria

These criteria are particularly important in selection of a DSS generator :

1. Data management functions
2. Model management functions
3. User interface capabilities
4. Compatibility and degree of connectivity
5. Available hardware platforms
6. Cost
7. Quality and availability of vendor support

24

DSS User Interface Issues

The unique characteristics of a DSS user interface stem from the unique characteristics of typical end users:

- They play an organizational role based on something other than computing skills.
- They have latitude in exercising judgment.
- Their decisions have impact.
- They spend more time on tasks that do not need a computer than do.
- The unique nature of the decisions they make means their personal preferences must be accommodated.

25

Factors Related to the Quality of the User Interface

- Learning curve – how fast does the user learn?
- Operational recall – how long does it take the user to recall how to use the DSS?
- Task-related time – how long is the typical task?
- System versatility – does it support a variety of end user tasks?

26

Factors Related to the Quality of the User Interface

- Error-trapping and support – what type of errors will users make?
- Degree of system adaptability – will it adjust to individual use?
- Management of cognitive overload – to what extent does the DSS reduce the need to remember things while using it?
- Degree of personal engagement – to what extent is the DSS enjoyable to use?
- Degree of guidance and structure – to what extent does the interface guide the user?

27

EXERCISE

- Observe a decision-making process in an organization you are familiar with. Assume that you are developing a DSS for this process. List your considerations for selecting the development tool(s).

28