Information Management

Munawar, PhD

2. System Development Methodology

Munawar, PhD

System Development Methodology

- A system development methodology refers to the framework that is used to structure, plan, and control the process of developing an information system.
- A wide variety of such frameworks have evolved over the years, each with its own recognized strengths and weaknesses.
- One system development methodology is not necessarily suitable for use by all projects.
- Each of the available methodologies is best suited to specific kinds of projects, based on various technical, organizational, project and team considerations.

Prototyping

- Not a standalone, complete development methodology, but rather an approach to handling selected portions of a larger
- Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process
- User is involved throughout the process, which increases the likelihood of user acceptance of the final implementation.
- Small-scale mock-ups of the system are developed following an iterative modification process until the prototype evolves to meet the users‘ requirements.
- While most prototypes are developed with the expectation that they will be discarded, it is possible in some cases to evolve from prototype to working system.
- A basic understanding of the fundamental business problem is necessary to avoid solving the wrong problem.

The Strengths of prototyping

- Addresses the inability of many users to specify their information needs, and the difficulty of systems analysts to understand the user‘s environment, by providing the user with a tentative system for experimental purposes at the earliest possible time.
- Can be used to realistically model important aspects of a system during each phase of the traditional life cycle.
- Improves both user participation in system development and communication among project stakeholders.
- Especially useful for resolving unclear objectives; developing and validating user requirements; experimenting with or comparing various design solutions; or investigating both performance and the human computer interface.
- Potential exists for exploiting knowledge gained in an early iteration as later iterations are developed.
- Helps to easily identify confusing or difficult functions and missing functionality.
- May generate specifications for a production application.
- Encourages innovation and flexible designs.
- Provides quick implementation of an incomplete, but functional, application

The Weaknesses of Prototyping

- Approval process and control is not strict.
- Incomplete or inadequate problem analysis may occur whereby only the most obvious and superficial needs will be addressed
- Requirements may frequently change significantly.
- Identification of non-functional elements is difficult to document.
- Designers may prototype too quickly, without sufficient up-front user needs analysis.
- Designers may neglect documentation, resulting in insufficient justification for the final product and inadequate records for the future.
- Can lead to poorly designed systems. Unskilled designers may substitute prototyping for sound design, which can lead to a —quick and dirty system without global consideration of the integration of all other components.
- Can lead to false expectations, where the customer mistakenly believes that the system is —finished when in fact it is not; the system looks good and has adequate user interfaces, but is not truly functional.
- Iterations add to project budgets and schedules, thus the added costs must be weighed against the potential benefits. Very small projects may not be able to justify the added time and money, while only the high-risk portions of very large, complex projects may gain benefit from prototyping.
- Prototype may not have sufficient checks and balances incorporated.

Appropriate Situation for Prototyping

- Project is for development of an online system requiring extensive user dialog, or for a less well-defined expert and decision support system.
- Project is large with many users, interrelationships, and functions, where project risk relating to requirements definition needs to be reduced.
- Project objectives are unclear.
- Pressure exists for immediate implementation of something.
- Functional requirements may change frequently and significantly.
- User is not fully knowledgeable.
- Team members are experienced (particularly if the prototype is not a throw-away).
- Team composition is stable.
- Project manager is experienced.
- No need exists to absolutely minimize resource consumption.
- No strict requirement exists for approvals at designated milestones.
- Analysts/users appreciate the business problems involved, before they begin the project.
- Innovative, flexible designs that will accommodate future changes are not critical.

Least Appropriate Situation for Prototyping

- Mainframe-based or transaction-oriented batch systems.
- Web-enabled e-business systems.
- Project team composition is unstable.
- Future scalability of design is critical.
- Project objectives are very clear; project risk regarding requirements definition is low

incremental

- A series of mini-Waterfalls are performed, where all phases of the Waterfall development model are completed for a small part of the system, before proceeding to the next increment; OR
- Overall requirements are defined before proceeding to evolutionary, mini-Waterfall development of individual increments of the system, OR
- The initial software concept, requirements analysis, and design of architecture and system core are defined using the Waterfall approach, followed by iterative Prototyping, which culminates in installation of the final prototype (i.e., working system).

The Strengths of incremental

- Potential exists for exploiting knowledge gained in an early increment as later increments are developed.
- Moderate control is maintained over the life of the project through the use of written documentation and the formal review and approval/signoff by the user and information technology management at designated major milestones.
- Stakeholders can be given concrete evidence of project status throughout the life cycle.
- Helps to mitigate integration and architectural risks earlier in the project.
- Allows delivery of a series of implementations that are gradually more complete and can go into production more quickly as incremental releases.
- Gradual implementation provides the ability to monitor the effect of incremental changes, isolate issues and make adjustments before the organization is negatively impacted.

Weaknesses of incremental

- When utilizing a series of mini-Waterfalls for a small part of the system before moving on to the next increment, there is usually a lack of overall consideration of the business problem and technical requirements for the overall system.
- Since some modules will be completed much earlier than others, well-defined interfaces are required.

Appropriate situation for incremental

- Large projects where requirements are not well understood or are changing due to external changes, changing expectations, budget changes or rapidly changing technology.
- Web Information Systems (WIS) and event-driven systems.
- Leading-edge applications

Least Appropriate situation for incremental

- Very small projects of very short duration.
- Integration and architectural risks are very low.
- Highly interactive applications where the data for the project already exists (completely or in part), and the project largely comprises analysis or reporting of the data.

Spiral

- Focus is on risk assessment and on minimizing project risk by breaking a project into smaller segments and providing more ease-of-change during the development process
- Each cycle involves a progression through the same sequence of steps, for each portion of the product and for each of its levels of elaboration, from an overall concept-of-operation document down to the coding of each individual program. (Boehm, 1986)
- Each trip around the spiral traverses four basic quadrants: (1) determine objectives, alternatives, and constraints of the iteration; (2) evaluate alternatives; identify and resolve risks; (3) develop and verify deliverables from the iteration; and (4) plan the next iteration. (Boehm, 1986 and 1988)
- Begin each cycle with an identification of stakeholders and their win conditions, and end each cycle with review and commitment. (Boehm, 2000)

The Strengths of Spiral

- Enhances risk avoidance.
- Useful in helping to select the best methodology to follow for development of a given software iteration, based on project risk.
- Can incorporate Waterfall, Prototyping, and Incremental methodologies as special cases in the framework, and provide guidance as to which combination of these models best fits a given software iteration, based upon the type of project risk. For example, a project with low risk of not meeting user requirements, but high risk of missing budget or schedule targets would essentially follow a linear Waterfall approach for a given software iteration. Conversely, if the risk factors were reversed, the Spiral methodology could yield an iterative Prototyping approach.

Weaknesses of Spiral

- Challenging to determine the exact composition of development methodologies to use for each iteration around the Spiral.
- Highly customized to each project, and thus is quite complex, limiting reusability.
- A skilled and experienced project manager is required to determine how to apply it to any given project.
- There are no established controls for moving from one cycle to another cycle. Without controls, each cycle may generate more work for the next cycle.
- There are no firm deadlines. Cycles continue with no clear termination condition, so there is an inherent risk of not meeting budget or schedule.
- Possibility exists that project ends up implemented following a Waterfall framework

Appropriate situation for Spiral

- Real-time or safety-critical systems.
- Risk avoidance is a high priority.
- Minimizing resource consumption is not an absolute priority.
- Project manager is highly skilled and experienced.
- Requirement exists for strong approval and documentation control.
- Project might benefit from a mix of other development methodologies.
- A high degree of accuracy is essential.
- Implementation has priority over functionality, which can be added in later versions.

Least Appropriate situation for Spiral

- Risk avoidance is a low priority.
- A high degree of accuracy is not essential.
- Functionality has priority over implementation.
- Minimizing resource consumption is an absolute priority

RAD (Rapid Application Development)

- Key objective is for fast development and delivery of a high quality system at a relatively low investment cost.
- Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process.
- Aims to produce high quality systems quickly, primarily through the use of iterative Prototyping (at any stage of development), active user involvement, and computerized development tools.
- Key emphasis is on fulfilling the business need, while technological or engineering excellence is of lesser importance.
- Project control involves prioritizing development and defining delivery deadlines. If the project starts to slip, emphasis is on reducing requirements to fit the deadline, not in increasing the deadline.
- Generally includes Joint Application Development (JAD), where users are intensely involved in system design, either through consensus building in structured workshops, or through electronically facilitated interaction.
- Active user involvement is imperative.
- Iteratively produces production software, as opposed to a throwaway prototype.
- Produces documentation necessary to facilitate future development and maintenance.
- Standard systems analysis and design techniques can be fitted into this framework.

The Strengths of RAD

- The operational version of an application is available much earlier than with Waterfall, Incremental, or Spiral frameworks.
- Because RAD produces systems more quickly and to a business focus, this approach tends to produce systems at a lower cost.
- Engenders a greater level of commitment from stakeholders, both business and technical, than Waterfall, Incremental, or Spiral frameworks. Users are seen as gaining more of a sense of ownership of a system, while developers are seen as gaining more satisfaction from producing successful systems quickly.
- Concentrates on essential system elements from user viewpoint.
- Provides the ability to rapidly change system design as demanded by users.
- Produces a tighter fit between user requirements and system specifications.
- Generally produces a dramatic savings in time, money, and human effort

Weaknesses of RAD

- More speed and lower cost may lead to lower overall system quality.
- Danger of misalignment of developed system with the business due to missing information.
- Project may end up with more requirements than needed (gold-plating).
- Potential for feature creep where more and more features are added to the system over the course of development.
- Potential for inconsistent designs within and across systems.
- Potential for violation of programming standards related to inconsistent naming conventions and inconsistent documentation.
- Difficulty with module reuse for future systems.
- Potential for designed system to lack scalability.
- Potential for lack of attention to later system administration needs built into system.
- High cost of commitment on the part of key user personnel.
- Formal reviews and audits are more difficult to implement than for a complete system.
- Tendency for difficult problems to be pushed to the future to demonstrate early success to management.

Appropriate situation for RAD

- Project is of small-to-medium scale and of short duration (no more than 6 man-years of development effort).
- Project scope is focused, such that the business objectives are well defined and narrow.
- Application is highly interactive, has a clearly defined user group, and is not computationally complex.
- Functionality of the system is clearly visible at the user interface.
- Users possess detailed knowledge of the application area.
- Senior management commitment exists to ensure end-user involvement.
- Requirements of the system are unknown or uncertain.
- It is not possible to define requirements accurately ahead of time because the situation is new or the system being employed is highly innovative.
- Team members are skilled both socially and in terms of business.
- Team composition is stable; continuity of core development team can be maintained.
- Effective project control is definitely available.
- Developers are skilled in the use of advanced tools.
- Data for the project already exists (completely or in part), and the project largely comprises analysis or reporting of the data.
- Technical architecture is clearly defined.

Least Appropriate situation for RAD

- Very large, infrastructure projects; particularly large, distributed information systems such as corporate-wide databases.
- Real-time or safety-critical systems.
- Computationally complex systems, where complex and voluminous data must be analyzed, designed, and created within the scope of the project.
- Project scope is broad and the business objectives are obscure.
- Applications in which the functional requirements have to be fully specified before any programs are written.
- Many people must be involved in the decisions on the project, and the decision makers are not available on a timely basis or they are geographically dispersed.
- The project team is large or there are multiple teams whose work needs to be coordinated.
- When user resource and/or commitment is lacking.
- There is no project champion at the required level to make things happen.
- Many new technologies are to be introduced within the scope of the project, or the technical architecture is unclear and much of the technology will be used for the first time within the project.

FIS (Functional Information System)

- Supports a functional area by increasing its internal effectiveness and efficiency.
- Functional information systems typically follow the organizational structure
- Functional information systems are typically focused on increasing the efficiency of a particular department or a functional area such as accounting, human resources, marketing etc

Strengths of FIS

- Typically focused on increasing the efficiency of a particular department or a functional area.
- Such systems, rather than aiding organizational performance will act as inhibitors to an organization's development and change.

Weaknesses of FIS

- Although they may support a particular functional area effectively, they may be incompatible to each other (NO interaction between internal systems).
- A process may involve more than one functional area
- Some Information Systems are cross-functional

DSS

- A Decision Support System (DSS) is a computer-based information system that supports business or organizational decision-making activities.
- DSSs serve the management, operations, and planning levels of an organization (usually mid and higher management) and help to make decisions, which may be rapidly changing and not easily specified in advance (Unstructured and Semi-Structured decision problems).
- Decision support systems can be either fully computerized, human or a combination of both.
- Decision support systems generally involve non-programmed decisions. Therefore; there will be no exact report, content or format for these systems.
- Reports are generated on the fly.

Attribute of DSS

- Adaptability and flexibility
- High level of Interactivity
- Ease of use
- Efficiency and effectiveness
- Complete control by decision-makers.
- Ease of development
- Extendibility
- Support for modeling and analysis
- Support for data access
- Standalone, integrated and Web-based

Characteristics of DSS

- Support for decision makers in semi structured and unstructured problems.
- Support for managers at various managerial levels, ranging from top executive to line managers.
- Support for individuals and groups. Less structured problems often requires the involvement of several individuals from different departments and organization level.
- Support for interdependent or sequential decisions.
- Support for intelligence, design, choice, and implementation.
- Support for variety of decision processes and styles
- DSSs are adaptive over time.

Benefits of dss

- Improves efficiency and speed of decision making activities
- Increases the control, competitiveness and capability of futuristic decision making of the organization
- Facilitates interpersonal communication
- Encourages learning or training
- Since it is mostly used in non-programmed decisions, it reveals new approaches and sets up new evidences for an unusual decision

Components of dss

- DBMS → internal data (TPS & MIS) & external data (newspaper, online data services etc)
- Model Management Systems → store & access models that managers use to make decisions. Such models are used for analyzing the financial health of an organization, forecasting demand of a product or service etc.

Types of DSS

- Status Inquiry System: helps in taking operational management level or middle level management decisions, for example daily schedules of jobs to machines or machines to operators.
- Data Analysis System: needs comparative analysis and makes use of formula or an algorithm, for example cash flow analysis, inventory analysis etc.
- Information Analysis System: In this system data is analyzed and the information report is generated. For example, sales analysis, accounts receivable systems, market analysis etc.
- Accounting System: keep tracks of accounting and finance related information, for example, final account, accounts receivables, accounts payables etc. that keep track of the major aspects of the business.
- Model Based System: simulation models or optimization models used for decision- making used infrequently and creates general guidelines for operation or management.

Classification of dss

- Text Oriented DSS: It contains textually represented information that could have a bearing on decision. It allows documents to be electronically created, revise and viewed as needed
- Database Oriented DSS: it contains organized and highly structured data.
- Spreadsheet Oriented DSS: it contains information in spread sheets that allows create, view, modify procedural knowledge and also instruct the system to execute self-contained instructions. The most popular tool is Excel.
- Solver Oriented DSS: it is based on a solver, which is an algorithm or procedure written for performing certain calculations and particular program type.
- Rules Oriented DSS: It follows certain procedures adopted as rules. Export system is the example.
- Compound DSS: It is built by using two or more of the five structures explained above

EIS

- Executive information systems are intended to be used by the senior managers directly to provide support to non-programmed decisions in strategic management.
- These information are often external, unstructured and even uncertain. Exact scope and context of such information is often not known beforehand.
- This information is intelligence based:

Advantages of EIS

- Easy for upper level executive to use
- Ability to analyze trends
- Augmentation of managers' leadership capabilities
- Enhance personal thinking and decision making
- Contribution to strategic control flexibility
- Enhance organizational competitiveness in the market place
- Instruments of change
- Increased executive time horizons.
- Better reporting system

Advatanges of EIS (cont'd)

- Improved mental model of business executive
- Help improve consensus building and communication
- Improve office automation
- Reduce time for finding information
- Early identification of company performance
- Detail examination of critical success factor
- Better understanding

Disadvantages of EIS

- Functions are limited
- Hard to quantify benefits
- Executive may encounter information overload
- System may become slow
- Difficult to keep current data
- May lead to less reliable and insecure data
- Excessive cost for small company

KMS

- A knowledge management system is not radically different from all these information systems, but it just extends the already existing systems by assimilating more information.
- As we have seen data is raw facts, information is processed and/or interpreted data and knowledge is personalized information.

What is knowledge?

- personalized information
- state of knowing and understanding
- an object to be stored and manipulated
- a process of applying expertise
- a condition of access to information
- potential to influence action

Sources of knowledge of an organization

- Intranet
- Data warehouses and knowledge repositories
- Decision support tools
- Groupware for supporting collaboration
- Networks of knowledge workers
- Internal expertise

Purpose of KMS

- Improved performance
- Competitive advantage
- Innovation
- Sharing of knowledge
- Integration
- Continuous improvement by:
  - Driving strategy
  - Starting new lines of business
  - Solving problems faster
  - Developing professional skills
  - Recruit and retain talent

Activities in KMS

- Start with the business problem and the business value to be delivered first.
- Identify what kind of strategy to pursue to deliver this value and address the KM problem
- Think about the system required from a people and process point of view.
- Finally, think about what kind of technical infrastructure are required to support the people and processes.
- Implement system and processes with appropriate change management and iterative staged release.

Thank You !

Munawar, PhD