IMPLEMENTING CASE TOOLS IN THE INTELIGENT TELECOMMUNICATION SYSTEMS

Bahador Ghahramani, Ph.D., P.E., CPE
College of Information Science & Technology
University of Nebraska at Omaha
Omaha, NE 68182-0392 (USA)
E-mail: bghahramani@mail.unomaha.edu
Phone: 402-554-3975, Fax: (402) 5543400

Azad Azadmanesh, Ph.D.
College of Information Science & Technology
University of Nebraska at Omaha
Omaha, NE 68182-0392 (USA)
E-mail: azad@unomaha.edu
Phone: 402-554-3975, Fax: (402) 5543400

ABSTRACT

This paper discusses an intelligent and Internet-based Telecommunication System Specification Model (TSSM) using Computer Aided Systems Engineering tools (CASE tools). TSSM implements CASE tools to mechanize its lifecycle development maintenance and integration process. This model is developed to improve the system analysts (SA) efforts in their design and development of major software and hardware initiatives. This model also improves the SA effectiveness by guiding them through the system's Lifecycle Development Process (LDP). The CASE tools are used to support, integrate, and monitor all LDP functions of the system.

Keywords: Telecommunication Systems Specification Model, Information Technology, Lifecycle Development Process, CASE tools, Systems Design and Development

1. INTRODUCTION

The primary purpose of the intelligent Telecommunication System Specification Model (TSSM) is to assist the Systems Analysts (SA) through various aspects of the development process such as monitoring the development lifecycle process, maintaining, and regularly upgrading the system. The TSSM is a modern model that implements Computer Aided Systems Engineering tools, or CASE tools for designing and developing systems. TSSM helps the SA to better understand the system's functions, applications, and information required to perform them by implementing the CASE tools. The model utilizes the CASE tools throughout the system's Lifecycle Development Process (LDP). The CASE tools make the LDP more efficient by reducing the development time and expenses. The CASE tools also provide representations of LDP functions and applications that reflect various scientific options and help the SA reengineer an existing legacy system. Implementing the CASE tools also assists the SA to identify major functions and applications of hardware, software, and interface as the system is being developed [1]. The TSSM CASE tools applications are divided into two components:

- *Component CASE tools:* provide an online method of monitoring LDP components, features, functions, products, and activities.
- *Integration CASE tools:* facilitates integration of the LDP common purpose components, features, functions, products, and activities.

The CASE tools provide online support for: developing query files, reengineering activities, designing standards, and performing continuous quality improvement activities. The TSSM CASE tools are divided into eight interdependent primary modules: Database Access Module, Repository Interface Module, Graphic Design Module, Text Definition Module, User Interface Module, Monitoring and Evaluation Module, Quality Control Module, and User Interface Module [2].

The TSSM CASE tools are far superior to other similar technologies because of their practical applications in industry. The most important characteristics of the TSSM CASE tools are that, through its Neural Network Technology (NNT), it builds within itself a design technology that is the driving force throughout the LDP. The CASE tools are generally divided into two categories: (1) Upper CASE tools supporting the LDP strategic planning, analysis, and monitoring activities; and (2) Lower CASE tools supporting the other LDP activities [3].

From the SA perspective, the TSSM improves the quality of LDP by implementing the CASE tools and using these tools for on line monitoring of development activities, identifying the missing components, reengineering the failed modules, and continuously controlling the process. The CASE tools display a high-level overview of the LDP; and integration of the system's hardware, software, and interface modules so that the SA can understand how their areas of responsibilities integrate with each other. The intelligent TSSM provides two primary functions [4]:

- *User intelligence:* provides a modern Internet-based process that allows the SA to separate and integrate different LDP activities effectively.
- *CASE tools intelligence*: provides an interface capability for both novice and experimental modes of operations by supporting both hybrid methodologies and logic applications.

2. RATIONALE

The primary rationale for implementing the CASE tools is that they encompass a cutting-edge-technology that is capable of increasing SA productivity during the LDP. Through applications of the Internet and the NNT technologies, the TSSM monitors information flow (e.g., voice and data) and interactions between information and various system modules [5].

The LDP starts with the users' requirements, or wish list, and evolves into the SA detailed specifications that explain the logic of hardware, software, and interface. The model uses an online and mechanized matrix cross referencing technique to illustrate the current LDP progress. The detailed specifications are then fed into the reference module for analysis and reengineering [6].

Another rationale for TSSM CASE tools is integration, linkage, and monitoring LDP activities at different levels. The TSSM is a conceptual and logical model that monitors information flow within the modules and how the CASE tools are implemented, and whether the logical levels of specifications are met. The CASE tools assist the SA in providing a reengineering technology that enhances the LDP progress and application quality. In TSSM, each CASE tool supports a designated LDP activity or a set of activities. Upper CASE tools are assigned to the initial LDP designing activities such as analysis, planning, and other related efforts. Lower CASE tools are assigned to later LDP activities such as prototyping, alpha and beta testing, reengineering, and other related efforts. The CASE tools also detail specific database design processes, screen and report prototyping, query file development, and a host of other LDP functions. In general, the Upper CASE tools support the logical design and the Lower CASE tools support the programming aspects of the LDP [7].

The TSSM integrates various LDP applications by implementing the CASE tools through the model's intelligent Knowledge Ware tool interface software. The Knowledge Ware software includes such software as Planning Workstation (PWS), Rapid Application Development (RAD), Design Workstation (DWS), Analysis Workstation (AWS), Documentation Tool (DOC), and Construction Workstation (CWS). The TSSM also uses other Knowledge Ware software for more specific projects and applications. The TSSM software CASE tools use the following definitions [8]:

- *Information:* voice and data files, databases, servers, and routers
- *Process:* functions, programs, practices, policies, and standards
- *Technology:* hardware, software, and interface

The TSSM CASE tools improve LDP quality by helping the SA minimize: (1) software, hardware, and interface quality inconsistencies, (2) integration and linkage problems, (3) user requirements and SA specifications incompatibilities, (4) project costs and expenses inconsistencies, (5) project delivery deadlines and deliverable inconsistencies, (6) alpha and beta testing standards inconsistencies, (7) lack of proper documentations, and (8) unachievable expectations, requirements, and specifications [9].

3. TECHNOLOGIES

The CASE tools environment provides a complete intelligent support system for the TSSM process beginning with the concept level analysis and working through to the maintenance and reengineering phases. The CASE tools are the primary foundation for the entire LDP supporting the LDP information logic flow throughout the model's architecture. The CASE tools also benefit various TSSM project management activities such as project planning and monitoring, applications development and definition, data analysis and normalization, database schema development, development of user requirements and SA specifications, development of bug-free code in the system's selected language, comparative analysis of the generated code with the SA specifications, and application logic. Figure 1 is a presentation of the TSSM technologies and structure [10].

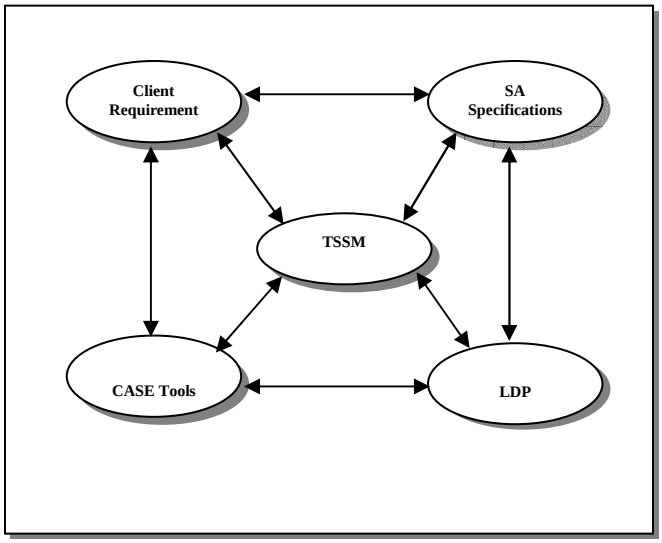

Figure 1: Telecommunication System Specifications Model Technologies and Structure.

The online CASE tools allow: (1) Internet-based customized documentation and reporting mechanism; and (2) intelligence on information quality control such as checking and cross-checking of the information flow through the TSSM for both accuracy and completeness. To improve efficiency of the LDP, the following modern technologies are incorporated into the TSSM CASE tools [11]:

- *Integration Analyzer:* provides an easy access to online Internet graphics, texts, and other formats. This technology integrates all LDP phases and applications. The integration activity is seamless because it is user-friendly, Internet accessible, and transparent to the users. It includes an intelligent online conversion of diagrams and design texts into other forms such as program codes, or encryptions. The integration covers all phases of the LDP activities, and helps the system to become multi-user and available through the Internet [12].
- *Artificial Intelligence Analyzer*: provides an effective method of self evaluation throughout the LDP. The technology provides a continuous quality control of the LDP by identifying and evaluating the information flow and comparing the results against a set of standards and specification. It also compares logical and physical designs of the LDP to determine variations and compatibilities. It measures completeness and consistency of LDP phases and activities at random intervals to reduce its complexity [13].
- *Static Code Analyzer*: analyzes the syntax and excitability of codes and specifications without executing the codes themselves by cross referencing the code references in software or a line of codes. The module identifies codes that are not executed or are part of infinite loops. It also determines the number of times data was executed and the frequency of the errors; and it performs fault analysis and other problem identification and resolution methods [14].
- *Dynamic Code Analyzer*: processes information as the LDP is being developed by tracking, evaluating, calibrating, checking, and identifying codes.

- *Coverage Analyzer*: analyzes and determines the level and degree that the LDP is processed using the test data.
- *Tracking Analyzer:* analyzes, monitors, and identifies key LDP variables, tracers, and statement codes through their execution paths.
- *Tuning Analyzer:* determines frequency of the LDP programs executed, identifies problem areas, tracks the problems, and activates the reengineering process.
- *Timing Analyzer:* determines the execution characteristics of the LDP such as time, frequency, date, duration, etc.
- *Resource Analyzer:* determines number of times a database transaction occurs; reports input and output execution times; and records hardware, software, and interface activation times.

4. BENEFITS

The primary benefit of implementing TSSM is monitoring test case application results (e.g., passed, failed, deferred, or not executed). This enables the SA to quantitatively and objectively monitor progress of a feature and compare its progress with users' requirements. This monitoring process is conducted throughout the software development cycle from the concept phase to the (final) product phase. The TSSM process also helps the SA review test case implementation results throughout the LDP cycles. The process highlights potential bottlenecks and identifies potential problem areas. After determination of potential problems, the SA devise proper courses of action to improve the system process [15].

CASE tools are used to minimize or eliminate potential LDP problems by initiating the following activities: (1) tracking feature development; (2) defining feature specifications and design requirements; (3) detecting potential problem areas; (4) evaluating feature thoroughness; (5) testing feature consistency; (6) identifying resources; (7) satisfying customer needs; (8) documenting results; and (9) implementing decisions and remedies [16]. The practicable benefit cycle of the CASE tools implementation is presented in Figure 2.

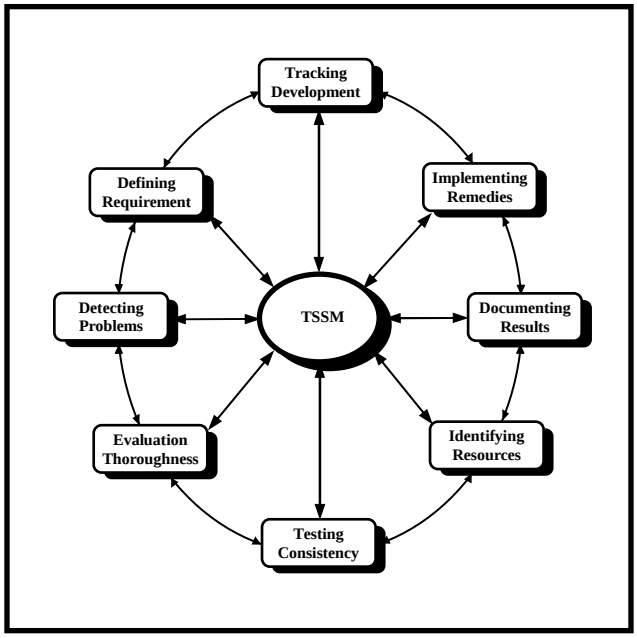

Figure 2: Telecommunication System Specification Model Cycle of Benefits.

The CASE tools most significant benefits occur during the maintenance phase of the LDP where monitoring the system operations become the most critical factor. The CASE tools have five online toolsets that significantly improve the SA activities: (1) designing toolsets that supports development of the deliverables and depends on the users requirements and SA specifications; (2) planning toolsets that prioritizes the deliverables based on their importance, functions, applications, and availabilities; (3) implementing toolsets that supports SA efforts in identifying and adapting new hardware, software, interface, and technologies; (4) maintaining toolsets that supports operations of the system after it is developed; and (5) storing toolsets in the Database Access Module that supports collection and organization of the development information [17].

The CASE tools effectively help SA by: (1) analyzing the hardware, software, and interface before their implementation; (2) evaluating the development process from top-down and bottom-up; (3) performing sensitivity analysis by activating run-test cases to answer "what if questions"; (4) developing scheduling techniques to coordinate the activities; (5) documenting and storing the relevant information in the Database Access Module; and (6) reengineering using compilers and assemblers to improve the activities [18].

5. CATEGORIES

The TSSM categorizes its CASE tools by their functions, applications, and plans. The function category includes: (1) user interface management; (2) configuration management; (3) modeling and simulation; (4) monitoring, estimation, and measurement; (5) prototyping; (6) test data generation, and (7) code generation. In addition, the above six characteristics are used to [19]:

- Access information from the Database Access Module;
- Provide an array of different query files; and
- Integrate and monitor the SA activities.

CASE tools integration serves three purposes: (1) integration of the individual CASE tools that are designed for different functions, (2) integration of each CASE tool with other LDP activities; and (3) integration of the CASE tools with the LDP strategic planning activities. The CASE tools application category greatly increases through [5]:

- *Productivity:* efficient allocation of resources; identification of hardware, software, and interface thresholds and staying within them; and monitoring and integrating lifecycle phases;
- *Quality:* continuously updating hardware, software, and interface;
- *Integration:* modernization of the hardware, software, and interface; and
- *Standardization:* adaptation of the industry standards and practices.

Successful LDP is possible when the CASE tools are supported by interdependent plans: (1) developing a structured lifecycle plan; (2) developing a formal information flow plan; (3) developing a continuous quality control plan for the hardware, software, and interface; (4) developing a high quality training plan for the SA and users; and (5) developing a plan to implement the Upper CASE tools and the Lower CASE tools [7].

6. TSSM MODULES

The modules discussed here are categorized as the primary TSSM modules and the CASE tools modules. The primary TSSM modules are the Management Module, the Linkage Module, and the Data Flow Diagram Module.

The Management Module ensures that the LDP activities conform to the SA specifications and industry standards. The Management Module tracks LDP activities into two levels: (1) monitoring progress of LDP activities and comparing the results with the development plans; and (2) monitoring the interdependencies of the activities. The updated SA development efforts are fed into the Management Module for further reviews and recommendations by the SA and users. The CASE tools facilitate the Management Module's monitoring and tracking activities by their functions, applications, and priorities. In addition, the Management Module provides online status reports of all LDP activities [15].

The Linkage Module integrates the TSSM modules and components with the CASE tools. The module's bridging process cross-references various activities with the SA detailed specifications files to: (1) identify common functions and applications; (2) verify the integration activities; and (3) document and update the results using the mechanized matrix approach. The online matrix also identifies, monitors, and documents relationships among various LDP activities and activates the Linkage Module through the following initiatives [13]:

- Position the development process to be used for competitive advantage
- Across the board consistency of the efforts
- Quality of the process
- Quality responsiveness to the users
- Reduce the development cost
- Ensure the effectiveness of the SA and the users
- Monitor the life- cycle phases
- Improve predictability of the process
- Optimize the maintainability and adaptability of the software, hardware, and interface
- Meet user requirements and SA specifications
- Satisfy the process goals and objectives

TSSM uses the Data Flow Diagram Module for viewing functional characteristics of the LDP activities. This module illustrates the system's architecture, design structure, and specifications to facilitate the LDP. It defines information (voice and data) flowing through the LDP, its impacts on various units, and the results. The TSSM Data Flow Diagram Module architecture is hierarchical and divided into the top level and the lower level that includes a context chart. The top level of the architecture is referred to as the context chart or the level zero design. The context chart is to define the external factors impacting the development process in detail and identifying the factors boundaries. The module also addresses two issues: (1) it identifies all possible scenarios and finite states an activity may follow, and (2) it identifies all activities that are taking place during the LDP phases [11].

7. CASE TOOLS MODULES

The primary CASE tools modules include the following online and Internet-based modules:

- *Database Access Module:* provides a variety of functions including object oriented lifecycle development, object oriented design with and without code support, object oriented coding with and without design support, and advanced object oriented application support. This module is used to develop the system's database architecture, physical definition from logical specifications, testing and maintenance requirements, and reengineering activities. In addition, it monitors relationships, interfaces, and related information [2]. Since the Database Access Module is one of the most important CASE tools modules, more detailed information is provided in the next section.
- *Repository Interface Module:* supports the definition of different types of hardware, software, interfaces, modules, and activities that are used throughout the LDP. It is a data repository and an active data dictionary.
- *Graphic Design Module:* supports the LDP engineering designs, drawings, layouts, and other related activities. It also evaluates quality of the LDP activities based on their predefined standards and specifications.
- *Text Definition Module:* supports the Database Access Module and the Repository Interface Module through definition of names, contents, and details of items in the two modules.
- *User Interface Module:* is the interpreter that determines the form and the format of both text and graphic information.
- *Monitoring and Evaluation Module:* is the intelligence behind the LDP. It analyzes all aspects of the LDP phases and determines whether they conform to the SA specifications and definitions, and if they are compatible with each other.
- *Quality Control Module:* supports and checks quality of all LDP activities. It continuously keeps track of the quality by comparing and mapping relevant activities with their pre-approved standards and specifications.
- *User Interface Module:* supports users and SA efforts from the system concept to its completion phases through online Internet processing and reporting software.

8. DATABASE ACCESS MODULE

The CASE Tools' Data Access Module software is used to provide LDP information to SA to develop new systems. The module is a comprehensive repository of all the data elements in the TSSM. Its main function is to evolve into the primary repository for project-unique definitions, acronyms, terminologies, and identifications. It categorizes the LDP activities into functions and applications. The module is also used as a repository of system specifications, source documents, query files, activities, data structure and architecture, algorithms, output processing, functions, and applications [8]. In most cases, the module is used to evaluate the users design requirements, SA specifications, process inputs, description of protocols, scaling, encryptions, rates, calibrations, and other input related information. It is designed to input characteristics of data dynamic and static elements. The data entries are online, are Internet-based, are supported by pre-designed templates, and are easily accessed through browsers. The module's data elements are embedded in the

program code, design software, and interface specifications [17].

The CASE Tools Database Access Module stores the information to support the following design features:

- *Diagram feature:* facilitates design, development, and reengineering parts of the system. This feature includes diagramming logic, and diagram clean-up logic that helps the SA to focus on the system specifications and requirements. This feature is also capable of identifying, monitoring, storing, and screening a diagram within the Database Access Module.
- *Data integrity feature:* ensures data integrity throughout the LDP. This feature also maps, evaluates, and compares SA specifications with their standards; and provides online recommendations.
- *Monitoring feature*: tracks reengineering activities of the LDP. This feature tracks each version of the process through user identifiers and provides an online file of the changes, updates and backups, revisions, and recommendations.

9. PROCESS

The LDP implements CASE tools to help SA develop an online, Internet-based, and intelligent program code from the specifications. CASE tools support the SA reengineering efforts through the SA detailed specifications document by: (1) identifying common functions among the LDP activities; (2) integrating similar activities, functions and applications; (3) integrating various hardware, software, and interfaces; (4) simplifying the activities; (5) developing and updating the working source codes; and (6) analyzing, screening, storing, and updating related information into the Database Access Module [14].

The LDP is an inclusive process that applies to system specification analysis, architecture, and software design specifications as well as reengineering the legacy systems. The process must also adhere to the following specifications:

- Prescribe specific guidelines for each notation within the process;
- Specify consistent relationships among notations;
- Provide the SA guidelines to reduce notation deviations from their norms;
- Verify the process through a structured reference practice; and
- Improve the system after implementation of an effective and tested computer-aided systems design and development CASE tools.

The TSSM process that satisfies the above specifications is referred to as the Structured Specification Process (SSP). This method is a step-by-step implementation technique that begins with the concept phase and ends with the completion phase [15]. Figure 3 is an implementation flowchart overview of the SSP method and steps. As this figure indicates, the SSP process initiates a series of steps from the Start phase to the End phase of the LDP. These TSSM steps must be performed for successful completion of the LDP. There are three primary TSSM methods currently being used: Structure Analysis and Design (SAD); Specification and Description Process (SDP); and state charts and related methods. Due to their technical implications, the SA mostly use the SAD and SDP methods, which are discussed here [19].

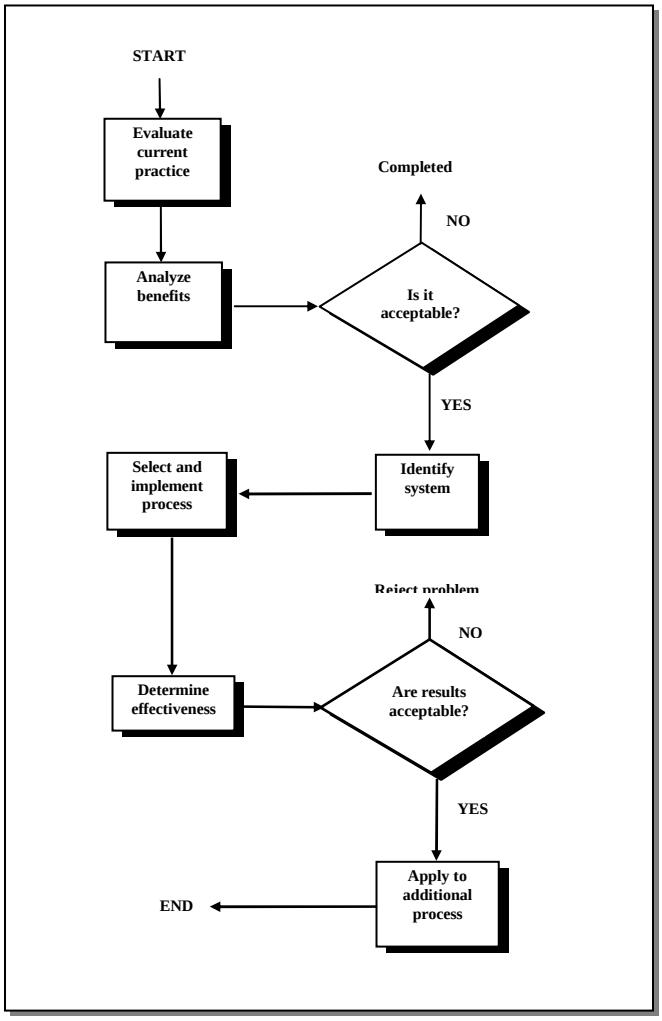

Figure 3: Flowchart Overview of the SSP Method.

10. STRUCTURED ANALYSIS AND DESIGN

The SAD method implements and evaluates a series of target system models instead of a single one during the LDP.

- *Requirement Models:* are used to depict the target system as a network of processing centers and are capable of transforming input information into expected data;
- *Architecture Models:* are similar to the requirement models except that data transformation and storing are interfaced to the processors, programs, and storage media upon which they will be considered; and
- *Design Models*: are created for each program in the target system. They view a target system's program as a set of functions and procedures that are interrelated.

The TSSM incorporates the above modules to develop data flow diagrams as master notations for its requirements. Figure 4 is a presentation of the TSSM and its features. Figure 4 also shows how information is processed and flow through TSSM and its features. The SAD method consists of the following modules [16]:

- *External Modules*: operate outside the target system. They have to interact among themselves on a regular

basis. These entities have dual functions; they can be used as sources as well as operating as sinks for information.

- *Data Processor Modules*: perform dual functions; they can be either primitive or higher level transformers.
  - Primitive transformers use pseudo codes to define their own behaviors.
  - Higher-level transformers permit development of a structured approach that represents system elements at various levels of detail and complexity.
- *Data Flow Modules*: direct information flow from one criterion to another and are typically shown by arrowheads. Each data flow is identified by a name that is entered in the data dictionary.
- *Data Store Modules:* store product data, and their characteristics and status.
- *Control Bar Modules*: permit modeling of control matters and their state transitions.

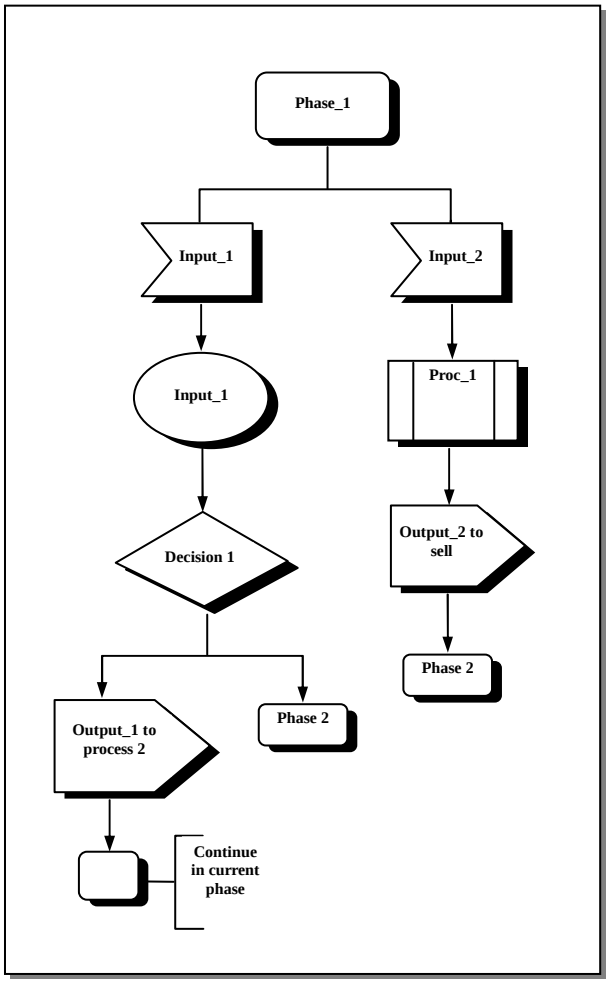

Figure 5: Flowchart Overview of the SDP/G Method.

11. SPECIFICATION AND DESCRIPTION PROCESS (SDP)

SDP methods apply two primary notations for realizing a system's requirements, architecture, and LDP phases. The first notation defines an individual entity in the system, and the second describes the methods used so that these entities can communicate. SDP methods differ from the SAD in that they are essentially independently implemented and are not impacted or influenced by architectural decisions. These methods recognize systems as sets of extended finite entities, communicating for a continuous period of time. An extended finite entity is defined as a system that processes data, defines the present state in any situation, and decides on an effective alternate course of action [17]. SDP flow charts are discussed and presented in two parts, which are independent and supplement each other. The first part is a graphic (SDP/G) presentation, and the second has a descriptive (SDP/D) format. Figure 5 is a flowchart description of SDP/G for the credit card example transaction in various phases of transaction. The corresponding SDP/D format, which is not shown here, is a descriptive presentation of the SDP/G. Following is a list of SDP definitions [16]:

- *Phase:* identifies the beginning and completion of a specific process;
- *Input:* specifies a particular phase—if a signal is input while the process is waiting in this phase, other actions are originated;
- *Output:* transmits signals to a specific recipient process;
- *Task:* activates an action to be taken;
- *Procedure:* activates a preprocessed action; and
- *Initiate:* activates a preferred course of action.

12. SYSTEM MONITORING

TSSM architecture allows easy access to the SA to continuously monitor the process and verify the accuracy of the results. Figure 6 is a presentation of the TSSM architecture. As Figure 6 indicates, there are key factors that are used to evaluate the effectiveness, efficiency, and applicability of the TSSM. If required, the factors are also used to perform task analysis and recommend reengineering remedies to improve the system or to prevent potential problems. As in any other software development procedure, most of the TSSM problems and bottlenecks are likely to occur in the beginning of the development process. This phenomenon is considered as a natural progression of a LDP, and requires sufficient resources to overcome the deficiencies [19]. The SA monitor TSSM developments and improvements regularly using the following key factors:

- *Feature problems:* decrease; functional feature problems found in the system test;
- *Emergency releases:* decrease; number of emergency releases incorporated into system test or field;
- *Degree of variations:* decrease; variation across features;
- *Test turnover decisions:* increase; decisions related to system test turnover;
- *Development resources:* decrease; resources of the development phase following turnover of the system;
- *Case numbers:* decrease; number of test specifications and test cases written between releases of the product;

- *Preparation times:* decrease; test inspection preparation time according to the established standards and norms; and
- *Delivery times:* decrease; test specification inspection delivery times.

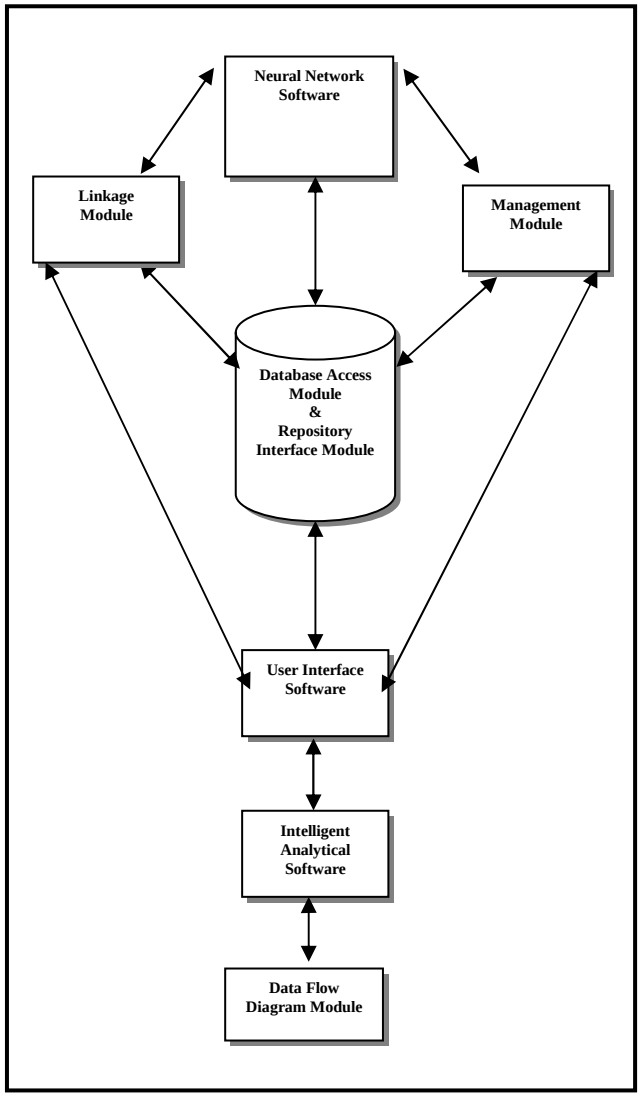

Figure 6: Telecommunication System Specification Model Architecture.

13. TEST SPECIFICATION

The TSSM test specification is a detailed formal documentation of all test cases that discuss and analyze a feature, a module, or an activity. The primary purpose of the specification is to test and evaluate TSSM features and operations of the software modules. Features could be internal or external in operations and applications. The internal features are the ones that are not visible to the users and are usually supporting the external modules and the services they provide. The external features are directly visible to the users and are primarily used for external operations and applications of the system [16].

There are two types of TSSM test specifications: Core Test Specifications and Elemental Test Specifications. The Core Test Specifications basically discuss features that provide new architectural elements that are essential in the LDP of the system and are on the critical path of the development cycle. They generally provide a common base for other feature developments. They are gauged by the SA to determine soundness of the system and to prevent potential problems. The Core Test Specifications are primarily in sequence, in that if one is disrupted, others are adversely impacted and a major part of the system may fail. These specifications must be gauged early in the LDP cycle and should be monitored throughout the process [15].

Elemental Test Specifications pertain to features that have a low degree of interdependencies with other features. Their failures, therefore, do not hinder the development progress and do not cause a major failure of the system.

14. SUMMARY

The primary goal of the intelligent and online TSSM is to design, develop, test, and maintain a telecommunication system using CASE tools. Through its NNT, the TSSM monitors and tracks all LDP activities. The Internet-based TSSM provides two functions: user intelligence, and CASE tools intelligence.

The CASE tools are implemented to improve the online LDP documentation, and validation of the SA specifications and user requirements. The LDP satisfies two objectives: (1) to finalize the development phase of the system, and (2) to decrease or eliminate the possible errors passing through the process. The LDP allows the SA to utilize more of their resources in successful completion of the system, and less on correcting errors, or solving problems [14].

The TSSM CASE tools help SA in two critical mechanisms: Internet-based customized documentation and reporting; and intelligence on information quality control for accuracy and completeness. The CASE tools are divided into two categories: Component CASE tools and Integration CASE tools. To support the LDP, the CASE tools utilize eight primary modules that are divided into two functional categories: Upper CASE tools supporting the LDP strategic planning, analysis, and monitoring activities; and Lower CASE tools supporting other LDP activities [13].

In summary, the TSSM LDP consists of four methodologies: implementing CASE tools, developing the system according to the SA specifications, testing the system, analyzing the test outputs, and monitoring the LDP. The TSSM benefits the SA efforts by effectively tracking the LDP activities, enabling the SA to quantitatively and objectively measure progress of the process, and comparing the results with the industry standards [18]. The LDP is an evolutionary as well as a revolutionary process that enabled the SA to make changes or modify the process whenever it is required.

15. ACKNOWLEDGEMENTS

The authors greatly appreciate our colleagues at the University of Nebraska at Omaha for their encouragement and support. This paper and project would not have been possible without generous grants from the NJK Holding Corporation and its subsidiaries. They are eternally grateful to Dr. Mark Pauley and graduate students Linfeng Cao, Louis Weitkam, and Megan Hrabanek of the University of Nebraska at Omaha for their efforts. In addition, our sincere gratitude is given to Systems Engineers in the Bell Laboratories and IBM Watson Research Center for their reviews and recommendations

16. REFERENCES

[1] Beath, C. M. & Orlikowski, W. J. "The Contradictory Structure of Systems Development Methodologies: Deconstructing the IS-User Relationship in Information Engineering." *Information Systems Research*, 10, 1994.

[2] Cerveny, R. P.; Garrity, E. J. & Sanders, G. L. "A Problem - Solving Perspective on Systems Development." *Journal of Management Information Systems*, Vol. 6, No. 4, 1990.

[3] Chechik, M., & Ding, W. "Lightweight Reasoning about Program Correctness." *Information Systems Frontiers*, Vol. 4, No. 4, 2002.

[4] D' Ambrogio, A., & Iazeolla, G. "Steps towards the Automatic Production of Performance Models of Web Applications." *Computer Networks*, Vol. 41, No. 1, 2003.

[5] Lending, D., & Chervany, N. L. "CASE tool Use and Job Design: a Restrictiveness/Flexibility Explanation." *Journal of Computer Information Systems*, Vol. 43, No. 1, 2002.

[6] Lundell, B., &Lings, B. "Comments on ISO 14102: the Standard for CASE tool Evaluation." *Computer Standards and Interfaces*, Vol. 24, No. 5, 2002.

[7] Insfran, E. & Pelechano, V. & Pastor, O. "Conceptual Modeling in the Extreme." *Information and Software Technology*, Vo. 44, No. 11, 2002.

[8] Baik, J., Boehm, B. & Steece, B. M. "Disaggregating and Calibrating the CASE tool varable in COCOMO II." *IEEE Transactions on Software Engineering*, Vol. 28, No. 11, 2002.

[9] Hong, S. S. "Schedulability Aware Mapping of Real Time Object Oriented Models to Multi Threaded Implementations." *Journal of KISS: Computing Practices*, Vol. 8, No. 2, 2002.

[10] Balaji, V. & Sangeetha, B. "Testing and Maintaining De-Localized Software Systems in a Multi Site Environment Using Web Based Tools." *Journal of Software Maintenance and Evolution Research and Practice*, Vol. 14, No. 3, 2002.

[11] Brinksma, E. "Verfication is Experimentation." *International Journal on Software Tools for Technology Transfer*, Vol. 3, No. 2, 2001.

[12] Pelechano, V., Pastro, O. & Insfran. "Automated Code Generation of Dynamic Specializations: an Approach Based on Design Patterns and Formal Techniques." *Data and Knowledge Engineering*, Vol. 40, No. 3, 2002.

[13] Keil, M. & Kendall, J. E. "Systems Analysis and Design." 3rd ed., Prentice Hall Englewood, New Jersey, 1994.

[14] Bradley, S. R. & Agoginto, A. M. "Design Capture and Information Management for Concurrent Design." Addison Wesley, New York, 1991.

[15] Raghunathan, B. & Raghunthan, T. S. "Adaptation of a Planning System Success Model to Information Systems Planning." *Information Systems Research*, Vol. 5, No. 3, 1994.

[16] Premkumar, G. & King, W. R. "Organizational Characteristics and Information Systems Planning: An Empirical Study." *Information Systems Research*, Vol. 5, No. 2, 1994.

[17] Short, J. E. & Venkatranan, N. "Beyond Business Process Redesign: Redefining Baxter's Business Network." *Sloan Management Review*, 1992.

[18] Chen,Y. M.; Miller, R. A. & Lee, D. L. "Object Oriented Part Model for Geometric Reasoning." *Journal of Integrated Computer-Aided Engineering*, Vol. 1, No. 5, pp. 375-395, 1994.

[19] Greenberg, S. "Real Time Groupware as a Distributed System: Concurrency Control and its Effect on the Interface." *Proceedings of the Conference on Computer Supported Cooperative Work*, Chapel Hill, North Carolina, USA, ACM Press, New York, pp. 207-217, 1994.

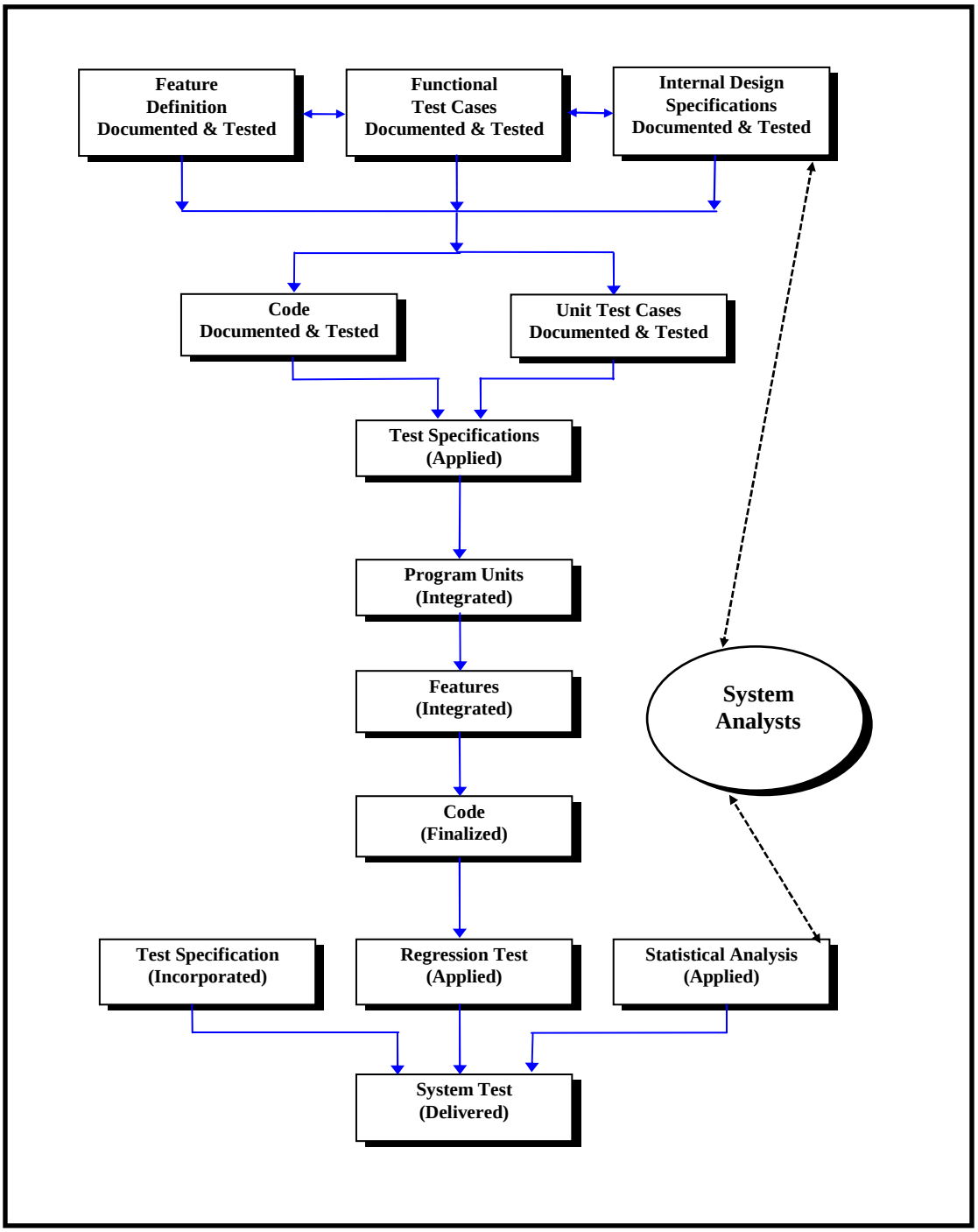

Figure 4: Information Systems Specification Model and Features.