Requirements-Driven ALN Course Design, Development, Delivery & Evaluation

Stephen J. Andriole
CIGNA

ABSTRACT

The best path to effective asynchronous learning network (ALN)-based course design, delivery and evaluation is through a requirements-driven methodology that recognizes the uniqueness of ALN-based learning. The methodology calls for the identification of purposeful and functional requirements, the identification of pre-course, early-course, mid-course and end-course activities, course "packaging" and prototyping, and "choreographed" delivery. It also calls for evaluation. The paper presents the methodology in the context of an actual course, a *Systems Analysis & Design* course offered asynchronously at Drexel University.

KEYWORDS

Design
Development
Delivery
Evaluation
Requirements

I. INTRODUCTION

This paper argues that the best path to an effective asynchronous learning network (ALN)-based course is through a requirements-driven discipline that recognizes the uniqueness of ALN-based delivery. The reason for the emphasis on requirements is simple: without reasonably accurate requirements, definitions and designs, we're likely to develop and deliver courses that might have elegant pedagogical features but little or no relationship to what students want or need.

The proposed discipline requires that we define and model requirements before we develop and deliver ALN-based courses. The assumption is that while conventional face-to-face (FTF) course design could well benefit from the discipline described here, given the nature of FTF delivery it's possible to significantly adapt to unanticipated events during a FTF course to maintain focus on the primary learning objectives. But on an ALN, that luxury is mitigated by asynchronous communications among the instructor(s) and students. Consequently, it's necessary to adopt more rigorous course requirements and design, development, delivery, and evaluation features that recognize the uniqueness of the ALN-based instruction. (See Fig. 1)

This paper describes the discipline and illustrates its application to a *Systems Analysis and Design* course delivered at Drexel University. The overall methodology assumes that front-end requirements analysis is an absolute prerequisite to successful design and development. It also assumes that the only way to improve the design and development process of ALN courses is to engage in systematic empirical evaluation of student judgments about the courses as well as how the courses actually generate desired learning outcomes.

The generic discipline stresses the common denominators of ALN-based design, development, delivery and evaluation. Some of these include the need for ubiquitous access to the network, absolute predictability, the need for network pedagogy anchored in an instructor-set "network personality," and the need to understand requirements prior to course design, all as Figure 1 suggests.

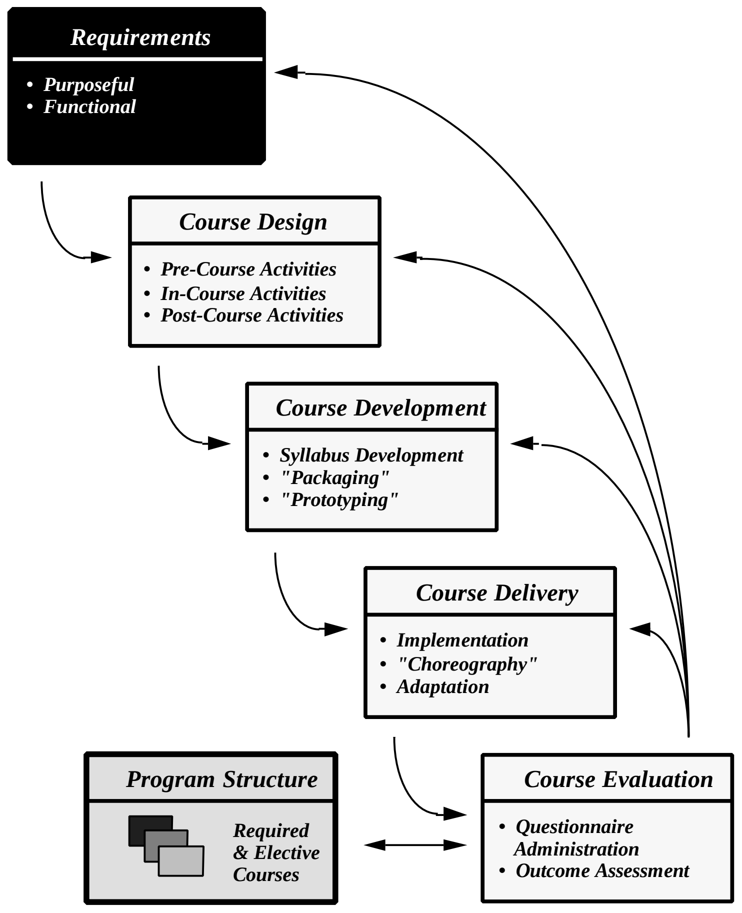


Figure 1: The ALN Course Design, Development & Evaluation Process

II. REQUIREMENTS

Requirements are the essence of successful course design, development, delivery and evaluation. Our breakdown recognizes **purposeful** and **functional** requirements.

A. Purposeful Requirements

The most important question, even before the requirements of specific learning modules, skills, and steps that lead to "competencies" is: **why** do we want the course in the first place? What longer term learning, understanding, and problem-solving objectives will be served by the course? How does it intersect with existing and planned courses? Most importantly, how will the course fit within whole programs of learning, such as degree and certificate programs?

Without answers to all of these questions, we run the risk of developing and delivering courses that are internally consistent but "disembodied" from larger objectives. Purposeful requirements analysis seeks to identify the reasons why a course exists and justify the additional time and effort necessary to define precisely how it should achieve these objectives. If it's difficult or impossible to identify and validate purposeful requirements then the course should at least be re-examined.

In any case, it's important that the results of the purposeful requirements analysis be documented -- as suggested in the following generic template (which is completed for the *Systems Analysis & Design* course).

Purposeful Objectives (for Systems Analysis & Design):	
Abstraction Objectives:	*To abstract the purpose of technology -- and specifically information systems -- in the larger business context; the ability to understand the driving forces behind the need for cost-effective information systems ...*
Synthesis Objectives:	*To integrate and synthesize larger technology issues, constraints and opportunities within the information systems implementation process; to appreciate the costs & benefits of information systems design, development and maintenance; to appreciate the hardware/software/communications intersection ...*
Course & Curriculum Linkage Objectives:	*To place the course in the larger context of related courses and whole programs; to see the linkages across courses (and disciplines); to appreciate the relationship among available courses, curricula, degrees and certificates ... to relate the learning objectives of Systems Analysis & Design to Data Base Management, User Computer Interface Design, Information Systems Implementation, Software Engineering, and Information Systems Evaluation ...*
Overall Understanding & Learning Objectives:	*To understand the systems analysis and design process in the larger systems engineering context; to understand alternative life cycle and system design and development process models; to understand the business value of information systems; to understand the processes by which cost-effective information systems are designed, developed, deployed and maintained ...*
Overall Interaction Objectives:	*To create an effective virtual learning environment; to demonstrate that topics-driven discussion "windows" and iterative model-supported design assignments support the learning process ...*

Table1: Purposeful Requirements Template

B. Functional Requirements

Once purposeful requirements have been identified and validated, functional requirements can be detailed. Functional requirements include the specific things the instructors and students expect the course to do. If purposeful requirements represent the 30,000 foot view of the course, functional requirements are at ground zero: they represent the specific things instructors want the students to know and be able to practice by the end of the course. Functional requirements represent the primary source of data for course design (See Table 2).

Functional Objectives: (for Systems Analysis & Design):	
Skills Objectives:	*To understand and be capable of implementing alternative life cycles, alternative requirements analysis and modeling methods, throwaway and evolutionary prototyping, demonstrating prototypes, evaluating prototypes, converting prototype specifications into software specifications, modeling software specifications (via alternative notation techniques and computer-aided software engineering [CASE] tools), managing the life cycle-driven process, and documenting the process ...*
Overall Competency Objectives:	*To understand how alternative life cycle steps can be combined, modified or eliminated to achieve cost-effective systems analysis and design; to prioritize requirements based on implementation and risk factors; to manage the whole process from ill-defined requirements to software specifications in the context of intersecting skills, disciplines and technologies ...*
Communications Objectives:	*To be able to communicate asynchronously individually and as a member of a project team ... to demonstrate competency using computer-based collaboration and modeling tools in a virtual space ... to express technical competency within a network via technical content, organized analyses and reports, and iterative commentary ... to develop network "identiites"*

Table 2: Functional Requirements Template

III. COURSE DESIGN

The essence of the design process is the conversion of requirements into a suite of tasks and activities that together constitute the course. We use a simple template for converting requirements into a set of "pre-course," "early-course," "mid-course," and "end-course" activities and tasks, and ALN interaction, data and software requirements.

This step should ideally be performed by instructors who have taught the course a number of times. While the instructors should work with general course "architects" (those with "domain-free" instructional design experience), the linkages among purposeful and functional requirements, and course learning tasks, should be validated by those with the widest and deepest domain in experience actually teaching the material being converted. This assumption contradicts some instructional design methods and approaches that assume that the conversion of learning requirements into instructional tasks can be performed adequately by those who may never have taught the material to be converted. Our experience suggests otherwise.

We also believe that ALN delivery affects the requirements to tasks conversion process. For example, learning tasks must be informed and contextualized in a virtual learning environment. This means that, for example, the task of **converting user requirements into exploratory prototypes** requires that students perform the requirements to prototyping conversion process asynchronously and collaboratively via a tool in the network. The task is therefore different and more complex than the same task would be in a FTF learning environment. We developed a template for converting requirements into tasks that recognizes that tasks to be completed in an ALN are different than those completed FTF. The difference here is not just "operational," that is, because success in an ALN depends on one's ability to work the technology; differences can also be traced to, for example, developing an interactive prototype that will be reviewed by the instructor and all of the students in the ALN.

Table 3 presents a snippet of the whole template for the *Systems Analysis & Design* course illustrating the segmentation of course activity, tasks and requirements.

The methodology requires that we think about the details of instruction within an ALN, not just the details of course delivery where much of the control over form, content, pace and feedback occurs FTF. Table 3 suggests how a specific course breaks down, but the methodology itself is generic.

Activities	***Tasks***	***ALN Interaction Requirements***	***Data Requirements***	***Software Requirements***
Pre-Course Activity				
Register &Students	Arrange Lotus Notes Training	Students Attend 1 Day Training on Lotus Notes	Training Materials	Access to LN Applications SoftwareActivity
Early Course Activity				
Face-to-Face Meeting	Explain Course to Students: Process, Scenario, Teams, Communi-cations Procedures	Classroom Setting	Course Materials to Students	Software Documentation
Mid-Course Activity				
Detailed Scenario Discussion	Scenario Features & Team Assign-ments, Including Especially Required Work Breakdown Structures	Faculty <---> Student Exchanges & Shared Team Materials, such as WBS, Plans & Schedules Via Conferencing Database	Course Materials Via Linked Documents	Student Access to Tools Database
End-Course Activity				
"Packaging" of Requirements Models, Proto-types & SRS ...	Conversion of Designs into Documentation; Inspect "Deliver-ables" ... ; Assess Trace-ability & Quality	Group (All Teams) Discussion of of Designs & Packaging (Docu-mentation) Pro-cess Via Document & Tools Linking Via Conference Database	Materials Designated by Team 1 & Other Teams' Suggest-ions; Additional Materials Sug-gested by Instructor Via Materials Data-base & Document Linking	Complete Access to Tools Via Tools Database

Table 3: Design of Systems Analysis & Design Course

IV. COURSE DEVELOPMENT

All of this permits:

- Development of the course syllabus
- Course "packaging"
- Course "prototyping"

The course syllabus is also organized around a template consisting of:

- Background information
- Course description
- Statement of course requirements
- Topics list
- Ways to communicate with the instructor and other students
- Course materials
- Detailed schedule of course events

Syllabus contents must link to purposeful and functional requirements and the activities matrix developed during the course design phase.

These features are important because they address many of the unique requirements of ALN-based teaching, specifically clear statements about what is expected of the students, ways to communicate and access materials, and -- especially -- the detailed, predictable schedule of topic "windows."

Following the development of the syllabus it's necessary to make sure that the whole course is "packaged" properly. This involves converting conventional materials (i.e., papers, textbooks, and presentations), putting them on the network, preparing to package the software applications necessary to support the interaction and communications processes, and making sure that everything works well together. Note that we assume that the preferred location of all course materials is the network and the students' PCs. All materials are thus available anytime and are local and network-accessible to all students and instructors. It's also possible, and often desirable, to add materials to an ongoing course: the quickest and easiest way to do so is to add the material to the network where it can be accessed and/or downloaded by students.

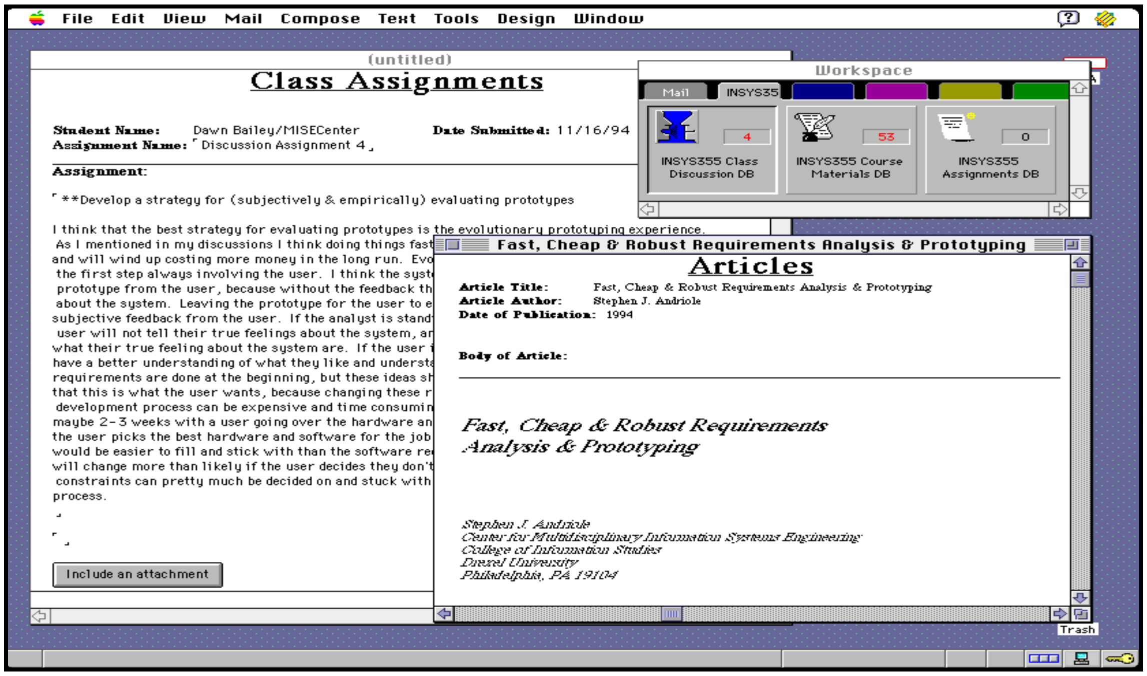
After packaging, it's necessary to "prototype" the course via a simulation of how the course should work. This process involves simulating:

- Access to the materials
- Asynchronous communications
- Threaded discussions
- Submission and critique of assignments
- Evaluation

Simulation is necessary due to the complexity of ALN-based instruction and our relative inexperience delivering ALN courses.

Figure 2 presents a simulation of how assignments, materials and the course "data bases" will interact within a Lotus Notes-based environment. This kind of simulation reduces uncertainty about reliability and robustness and also familiarizes those who will be required to support the course with the interaction process long before the course launches. In our case, the groupware in which we offer courses is based in Lotus Notes. We therefore develop simulations of how the course will actually "perform" from the Lotus Notes application that is the course itself. The simulation is an interactive "walkthrough," where the actual instructors and mock students interact on the network just as they will when the course goes "live."

Prototyping via simulation is ultimately a risk analysis and risk management process: if no problems are encountered (a rare occurrence) then the course can go to "production," but when problems are discovered the prototype permits iteration over time to correct the problems. Unlike a FTF course where in-person apologies can be extended when things go wrong, ALN courses and students have limited capacities for forgiveness.



(untitled)

Class Assignments

Student Name: Dawn Bailey/MISECenter **Date Submitted:** 11/16/94
Assignment Name: Discussion Assignment 4

Assignment:

**Develop a strategy for (subjectively & empirically) evaluating prototypes

I think that the best strategy for evaluating prototypes is the evolutionary prototyping experience.
As I mentioned in my discussions I think doing things fast
and will wind up costing more money in the long run. Evo
the first step always involving the user. I think the syst
prototype from the user, because without the feedback th
about the system. Leaving the prototype for the user to e
subjective feedback from the user. If the analyst is stand
user will not tell their true feelings about the system, ar
what their true feeling about the system are. If the user
have a better understanding of what they like and understa
requirements are done at the beginning, but these ideas sh
that this is what the user wants, because changing these r
development process can be expensive and time consumin
maybe 2-3 weeks with a user going over the hardware an
the user picks the best hardware and software for the job
would be easier to fill and stick with than the software re
will change more than likely if the user decides they don't
constraints can pretty much be decided on and stuck with
process.

Fast, Cheap & Robust Requirements Analysis & Prototyping

Articles

Article Title: Fast, Cheap & Robust Requirements Analysis & Prototyping
Article Author: Stephen J. Andriole
Date of Publication: 1994

Body of Article:

Fast, Cheap & Robust Requirements Analysis & Prototyping

Stephen J. Andriole
Center for Multidisciplinary Information Systems Engineering
College of Information Studies
Drexel University
Philadelphia, PA 19104

Figure 2: Simulated Lotus Notes-Based Display

V. COURSE DELIVERY

The delivery process consists of the full-scale "live" implementation of a course choreography that can adapt to some significant number of unanticipated events. It is essential that courses be "choreographed" to stage and anticipate events during the course: if course design is the substance of a course then choreography is its style.

Choreography requires that we think about roles and adaptive procedures, and there
are a variety of roles that the players (instructors, support staff and students) play during the course design, development, delivery and evaluation process. Here are the roles:

A. For Instructors

Initiator of discussions
Tutor of basic principles and methods
Impresario of student-to-student discussions
Problem-solving mentor

Router of questions to students, materials, etc.
Problem identifer/example setter
Tie-breaker
Grader
Humorist
“Therapist”
Team builder

B. For Support Staff

Student-to-student interaction supporter
Student-to-materials linking supporter
Routing supporter (to requested materials)
Student data base maintainer
Organizer of student grades

C. For Students

Participant in discussions
Submitter of assignments
Enhancer of ALN environment
Student-to-professor communicater
Student-to-student communicater

These roles suggest the kinds of behavior required of the players to make courses successful. They also suggest the kinds of simulations that can be run to prepare instructors for what may happen during an ALN course.

Adaptive in-course behavior requires that contingency plans be developed to deal with the following kinds of events:

- Students who attempt to dominate the course
- Students who ask trivial questions
- Materials that fail to “connect” with the students
- Interaction volume that exceeds expectations
- Communications problems
- Software problems
- Hardware problems
- Problems with the student teams
- Instructor and/or student illness

These and related problems can chain react in a lot of directions. It’s important to anticipate such problems and have contingency plans ready. Over time, a data base of problems/contingency plans can be developed so that solutions can be reused.

VI. COURSE EVALUATION

Without an evaluation it’s impossible to understand the immediate or longer-term effect the course is having on instructors, support staff or students. We have developed a questionnaire that measures student perceptions of how well (or poorly) the course was received. We have also developed a quality assurance process that compares student assignments generated in FTF courses and ones generated via ALN courses. The questionnaire measures perceptions across a variety of areas, some intentionally designed to compare FTF with network-based learning. A snapshot of responses -- within the context of some access and interaction data -- for the *Systems Analysis & Design* course appears in Table 4.

In addition to questionnaire data, we've developed a QA approach that compares and contrasts FTF and ALN assignments. We use the following measures:

- Quality of requirements models
- Quality of prototypes
- Quality of prototype evaluation
- Quality of software specifications
- Quality of documentation
- Quality of teamwork
- Overall creativity
- Ability to use design tools
- Timeliness

These measures permit comparisons with conventionally-delivered course products and outcomes and those generated via an ALN course. On a 3X3 comparison (3 conventional and 3 ALN *Systems Analysis & Design* courses taught by the same instructor), the ALN students always performed as well and often performed better than their conventional counterparts on all of the measures. Prototype quality was consistently higher in the ALN course than for the conventional course.

Results for 17 courses and 207 students:

Average # of Interactions (for Approximately 8 Weeks)
750 Interactions (for Average Course of 10 Students)

Preferred Interaction Times of Day
1. 8 PM - 12 Midnight (36% of Total)
2. 4 PM - 8 PM (22% of Total)
3. 12 Midnight - 4 AM (14% of Total)
4. 4 AM - 4 PM (28% of Total)

91% would take another ALN course

97% felt they had **more access** to the instructor than in "conventional" course delivery

80% felt that conventional courses were **more boring** than the ALN course

67% felt they had **more communication** with fellow students than in conventional courses

66% felt they **learned more** on the ALN-based course than they would have expected to learn in a conventional course

99% felt that seeing the ideas & assignments of others was useful.

Table 4: Interaction & sample Subjective Evaluation Data

VII. PROGRAM STRUCTURE

The final step is to look at evaluation results in the context of larger program directives. For the *Systems Analysis & Design* course the context is the masters degree in information systems. Individual courses must obviously "fit" within a larger program objective and the evaluation of the course's effectiveness should extend to assess the role the course plays in the degree (or certificate) program of which it is a part. "Core" and "required" courses should be assessed differently than "elective" courses, since required/core courses are required to help cumulate and synthesize knowledge and skills.

VIII. EMERGING FINDINGS, NEW REQUIREMENTS

This paper proposes that a requirements-driven discipline drive the ALN course design, development, delivery and evaluation process. We have had success with the discipline which represents a kind of "standard" template. We've learned that course design and development variation result in the antithesis of common-look- and-feel, a goal we believe reduces training time and costs -- and limits unnecessary debates about how to design, development, delivery or evaluation processes should proceed.

We've also made some inferences from our experiences that have helped, and will continue to help, enhance ALN discipline. It appears, after several course deliveries, questionnaire data analyses, and outcome assessments, that the following findings may well prove true over the long haul, findings that will help us enhance the course design, development, delivery and evaluation process:

- There is an enormous need for structure in an ALN environment

    1. Opening & closing discussion windows
    2. Clear discussion topics/readings/assignments schedule
    3. Completely predictable course schedule
    4. All materials online & accessible
    5. Common course "look & feel" -- especially in a multiple course ALN
    6. Real-time monitoring of student performance

- It can be made very cost-effective

    1. Hypothesized (maximum) quality ratios:
        - 1 instructor for every 30 students
        - 1 instructor + 1 ALN assistant for every 50 students
        - 1 instructor + 2 ALN assistants for every 75 students ...
    2. The methodology is repeatable
    3. The communications/hardware/software is off the shelf & relatively inexpensive -- and on the right price/performance trends ...

- Industrial information and software systems design processes can be "simulated"via ALNs

- We can design a learning process & an interactive asynchronous learning environment that accommodates self-pacing

- Computer-aided software engineering (CASE) and other tools can be integrated into a groupware environment

We are convinced that discipline in the form of predictability, consistency and reusability will pay dividends as we extend the reach of ALN education and training. The requirements have already led to the deployment of a methodology for design, developing, delivering and evaluating ALN courses. Over time, additional requirements will be validated and addressed during the design, development, delivery and evaluation processes. Hopefully, our methodology will grow wider and deeper over time.

About the author

Stephen J. Andriole is Chief Technology Officer and Senior Vice President for Technology Strategy at CIGNA Corporation, a global insurance and financial services organization. The work described in this paper is supported by The Alfred P. Sloan Foundation. The Sloan Program Officer is Dr. A. Frank Mayadas. The Sloan Foundation has an on-going research and development program in Asynchronous Learning Networks. We gratefully acknowledge the support of The Sloan Foundation without whose support this work would not have been possible.