Give the Customer What They Meant to Ask For

…or…

Why Do We Call It Instructional Systems Design?

By Peter R. Hybert

Analysis Was Always the Cuter One

Any newcomer to the field of training and/or performance technology is bound to notice that all anyone ever wants to talk about is analysis (or is it "e-nalysis" now?). The array of different types of analysis—needs assessment, audience analysis, performance analysis, knowledge/skill analysis, goal analysis, meta analysis—can be baffling to clients and practitioners. The worst thing you can do is "jump to solutions before understanding the problem." And, people must act on the belief that analysis is important or there wouldn't be a catch phrase to describe an all-too-common client complaint—"analysis paralysis."

Of course, analysis is important, but it is only part of the picture. Through analysis you *understand* the problem, the situation, the implications. Design is where you make critical decisions about how to best *address* the problem, situation, and implications given the complex set of stakeholder requirements, available resources, and environmental constraints that govern a specific project.

The most effective approach is to do *sufficient analysis* to make preliminary design decisions. Then conduct additional *focused analysis* as needed for more detailed design decisions, and so on, to eventually "spiral" to a solution that can then be implemented so it can then, finally, produce the *real end goal:* performance to a standard. *Analysis and design are both only means to reach the real end goal.* And, it is in nobody's interest to take too long to reach the real end goal.

Design, though, is the immediate "end in mind" for analysis. If analysis is breaking something into its parts to understand it better, design is synthesis, putting parts together to create an effective whole. With an instructional system, as in product development, design is where you should resolve significant show-stopping issues before moving to development, where the real money is spent.

Most writing about instructional systems design targets the more detailed levels of design—writing objectives, selecting media, laying out visuals, etc.—which is really instructional *materials or message* design. Actually, much of it is really development.

Why Design? Why Not Just "Rapid Prototype"?

New "rapid prototyping" approaches often skip the design step altogether (and sometimes analysis as well). Viewed in the best possible light, the idea is to get the customer something they can react to so the creator of the solution can get directed more quickly to a good solution. Too often the reality is that, without using a design process, the prototyping session is a little like trying to build something in the dark—you might hit on something good . . . but you might also saw through your coffee table.

Why Not Just Call a Course, a Course, Instead of Trying to Make It into an Instructional "System"?

Even a "one-off"—a single instructional product—is a system. But a larger view of an instructional *systems* design would encompass the overall curriculum, the delivery infrastructure, evaluation processes, etc. The collection of these processes makes up a system that takes inputs (students and information) and puts them through processes (instruction, assessment, and coaching/feedback) to generate an output (a learner with the intended new capabilities). Just calling it "training" makes it too easy for people to think of school rather than considering other ways to generate the desired output that may be more reliable or cost-effective than a traditional class.

Designing any system requires more than one level of design. In our model there are three—similar to other design processes, such as product development and building design. Table 1 describes three levels of design using nontraining examples.

Levels	Building Example	Product Example	Software Example
System (Macro) Design	Campus layout or map of the subdivision	Product line architecture	System architecture
Subsystem/ Component (Mid) Design	Plans for a single building, including structural plans and electrical wiring diagrams	Product specifications and drawings	Flow chart for a single application, dataflow diagram
Detailed/User Interface (Micro) Design	"Details" or "typicals" (that show "close-ups" of features/ steps that are used repeatedly (such as how exterior sheathing is attached to the building frame)	Individual part drawings and specs	Prototypes of screen layouts

Table 1: Nontraining Examples of Three Levels of Design

How Is Instructional Design Like Other Design Processes?

Design is one of those topics that people enjoy arguing about. Maybe because design seems a little like an art—there is no one right answer to a given design problem. As with art, there are people who come up with solutions others never dreamed of but which are readily recognizable as outstanding by those "in the know." (And, like art, many customers of instructional design *cannot* always tell a good design from a poor one.) With all that subjectivity, there is plenty to argue about.

Based on observation and analysis of the job performance of various types of designers (including software designers, product designers, art designers, building designers, and others), I think design in general shares the following common characteristics.

- *Design should start with requirements and needs*. You first look at the need and then create a solution to fit the need. It is not like consumer sales in which you try to create a need or desire for an already existing end product. Instead, you focus on customer functionality requirements. What does the customer need to accomplish with the object I'm designing? How might I design this product so that it can do that? Of course, the requirements should support a defined business goal.

- Design involves "*tradeoffs*." You can't always meet all the requirements, so you have to evaluate different solutions and decide which best meets the key requirements. Good designers have a "knack" for minimizing tradeoffs and, more often, for finding the *right* tradeoffs—the ones the customers will *agree* on and that history will show are the right tradeoffs.

- Design is *customer-oriented*. Solutions have to meet *perceived customer needs* and expectations as well as the technical specification. The solution really has to work *and* the customer has to like it. You either have to engage the customer in the design process or sell them on your solution after it is designed—usually, a little of both.

- Design is a *learning process* and, as a result, is (at least somewhat unavoidably) iterative. Nobody likes rework. But, if you are a designer, you have to get used to a certain amount of it—there is no way for you as a designer to fully understand all of your customer's requirements immediately. More importantly, your *customer* will not understand their *own* requirements or the impact of their decisions early in the design process. As an experienced systems architect once told me, "Customers expect you to give them what they *meant* to ask for."

- Design is an *individual thing*. A group cannot design; only an individual can design. Design requires someone to synthesize all the requirements and preferences and define a solution or an idea. A group can help evaluate it, tweak it, reject it, or even help explain it, but they cannot very easily do the initial conceptual design. Even when you use a team in the design processes, there still has to be a lead designer—the team's role is to *influence* the designer. And, if the design is simple, the above is much easier. If it is complex by necessity, this is much more challenging. A team may work on the design, but there has to be one overall designer if you want an integrated end product.

- Design requires a *broad to narrow focus*, from concept to specifications to designs to prototypes, etc. Some models show the process as spiral. However, most design models show a series of steps progressing from macro to micro with the tacit understanding that the process is not quite that neat and linear in real life. (See Figure 1 for two contrasting ways of illustrating a design process.)

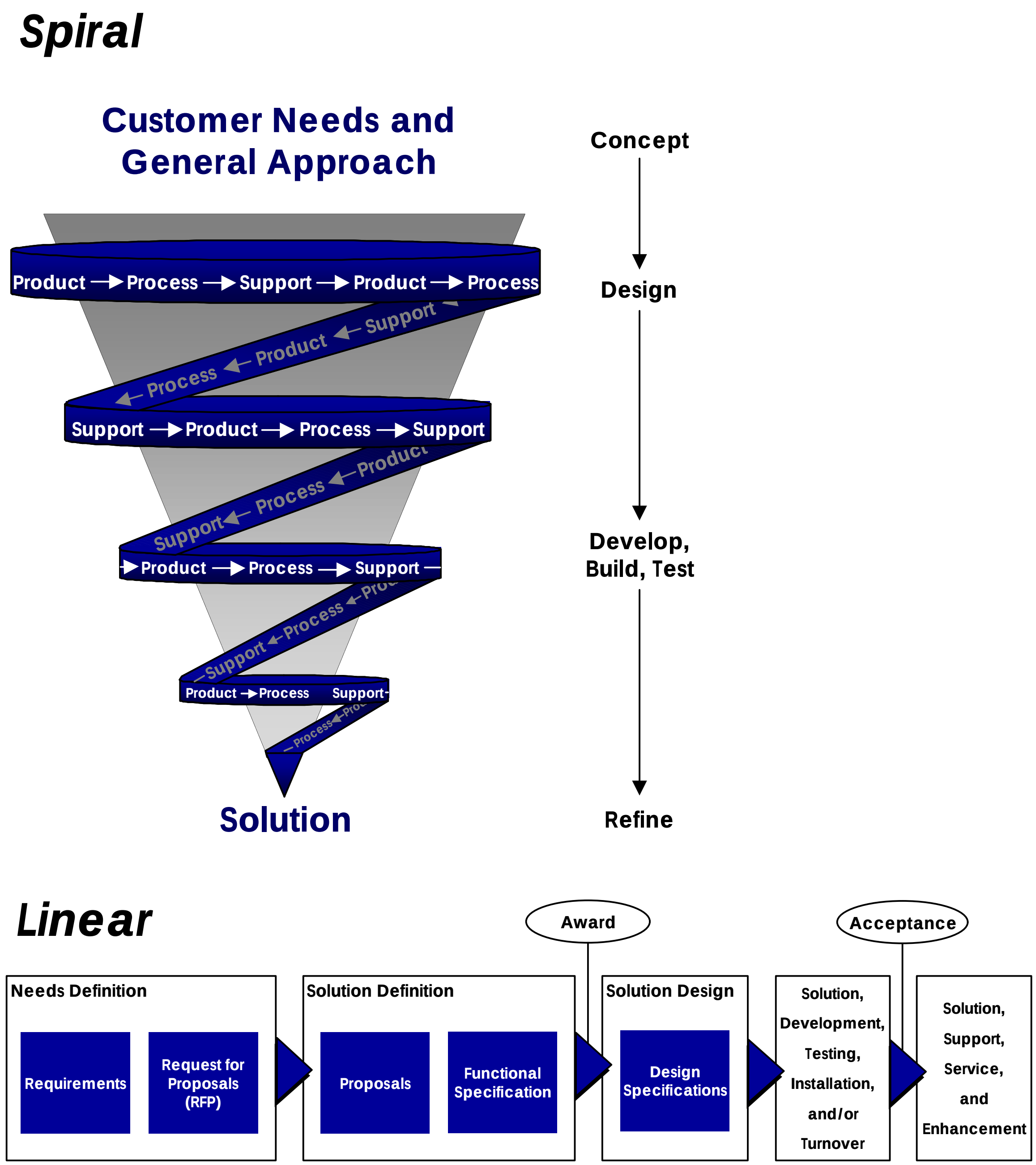
Spiral
Customer Needs and General Approach
Product → Process → Support → Product → Process
Process ← Product ← Support
Support → Product → Process → Support
Support ← Process ← Product
Product → Process → Support
Support ← Process ← Product
Product → Process Support
Process ← Product
Solution
Concept
Design
Develop, Build, Test
Refine
Linear
Award
Acceptance
Needs Definition
Requirements
Request for Proposals (RFP)
Solution Definition
Proposals
Functional Specification
Solution Design
Design Specifications
Solution, Development, Testing, Installation, and/or Turnover
Solution, Support, Service, and Enhancement

Figure 1: Two Contrasting Ways of Illustrating a Design Process

Most importantly, to get good designs quickly all (or most) of the time, you need to use a structured process and work systematically—it won't just evolve.

How Can You Tell a Good Design from a Poor One?

The classic engineering perspective is that "elegance" is the pure goal for design. Elegance implies a few key characteristics.

- Innovation/novelty
- Economy/simplicity (not complicated)
- Fit with/solution for the specific problem
- Robustness (works well in a wide range of situations)

The problem with elegance alone as the goal is that it is too narrow. It is based on the needs of only one stakeholder group—the design community. There are a number of stakeholders in every design situation. Table 2 illustrates four common categories of stakeholders.

Stakeholder Group	Example for a Training System or Course	Typical Requirements/Areas of Concern
Customers/users	• Learners • Their managers • Internal sponsor(s)	Ease of use, fit with immediate needs
"Owner"/manager of end product over time	Training department manager	Minimum revisions/changes, ease of making updates, fits various work applications/situations
"Implementer"	Trainer	Minimum labor overhead/minimum nonvalue effort to deliver, acceptability to end users
Far customer	Shareholders/business management	Payback/ROI

Table 2: Four Common Categories of Stakeholders

A concept used in product design processes is the series of "design-fors." It began as "DFM" or "design for manufacturability" but has expanded to "DFX," which might as well stand for "design for everything and everyone you can think of." Design for installability, design for serviceability, design for upgradeability, design for "distributability," design for maintainability, etc. This is really just another way of expressing the stakeholder concept above. DFX means understanding all the stakeholders *and* their application of/involvement with the product, and then designing the product to facilitate their use of it.

When you are designing training, there are a number of specific stakeholders within the stakeholder groups above. You have learners and their supervisors who make up the primary customer base.

They are equivalent to end users of a product. But, you also have a number of stakeholders that make up the supply/distribution system including instructors, learning coaches, administrators, network administrators (for online distribution), evaluators, etc. And you have the business stakeholders—the executives who expect a return on their training and development investment. Each stakeholder group has different requirements to be addressed at different levels of the design process.

The key to getting good designs on a regular basis is having the right stakeholders provide the right input at the right points in the process so you can design to meet their requirements, rather than achieve them through rework.

Three Levels of Training and Knowledge Management System Design

Using either the spiral or linear design approach, there are three levels of training and knowledge management system design. Focusing on each level reduces the number of decisions to be made at one time and accelerates the design and development of the content.

Levels	Training Example	Notes
System (Macro) Design	Plans for the content inventory, audience segment learning paths, evaluation system, delivery system, etc.	
Subsystem/ Component (Mid) Design	Plans for the content and instructional process for a single program or deliverable (e.g., course)	This is the level where most instructional design models begin
Detailed/User Interface (Micro) Design	Plans and prototypes of layout, format, “look and feel” standards	

In any design situation, you can begin at any one of these three levels; however, the ideal sequence would be to design the overall instructional system architecture before designing the instructional process for specific components within that system, and then finally the specific subcomponents of each component (at a very detailed level). In other words, design the curriculum and infrastructure first, the sequence of learning process(es) within each course or deliverable next, and activities and materials formats, layout, organization, etc., last at the user interface/instructional activity level.

Each level of design has different outputs and quality/success measures. In a series of follow-up articles, I have begun to identify key design goals for each level of design. The design goals are the requirements the designer should be trying to address. They are also a means for the design customer/stakeholder to evaluate the suitability of a given design. My intent with this series is to demystify the instructional design process while, at the same time, illustrate some of the challenges that make it a rewarding profession. I would be very interested in hearing your reactions.

Note: An earlier version of this article was published in the October issue of *Performance Improvement,* the journal of ISPI.