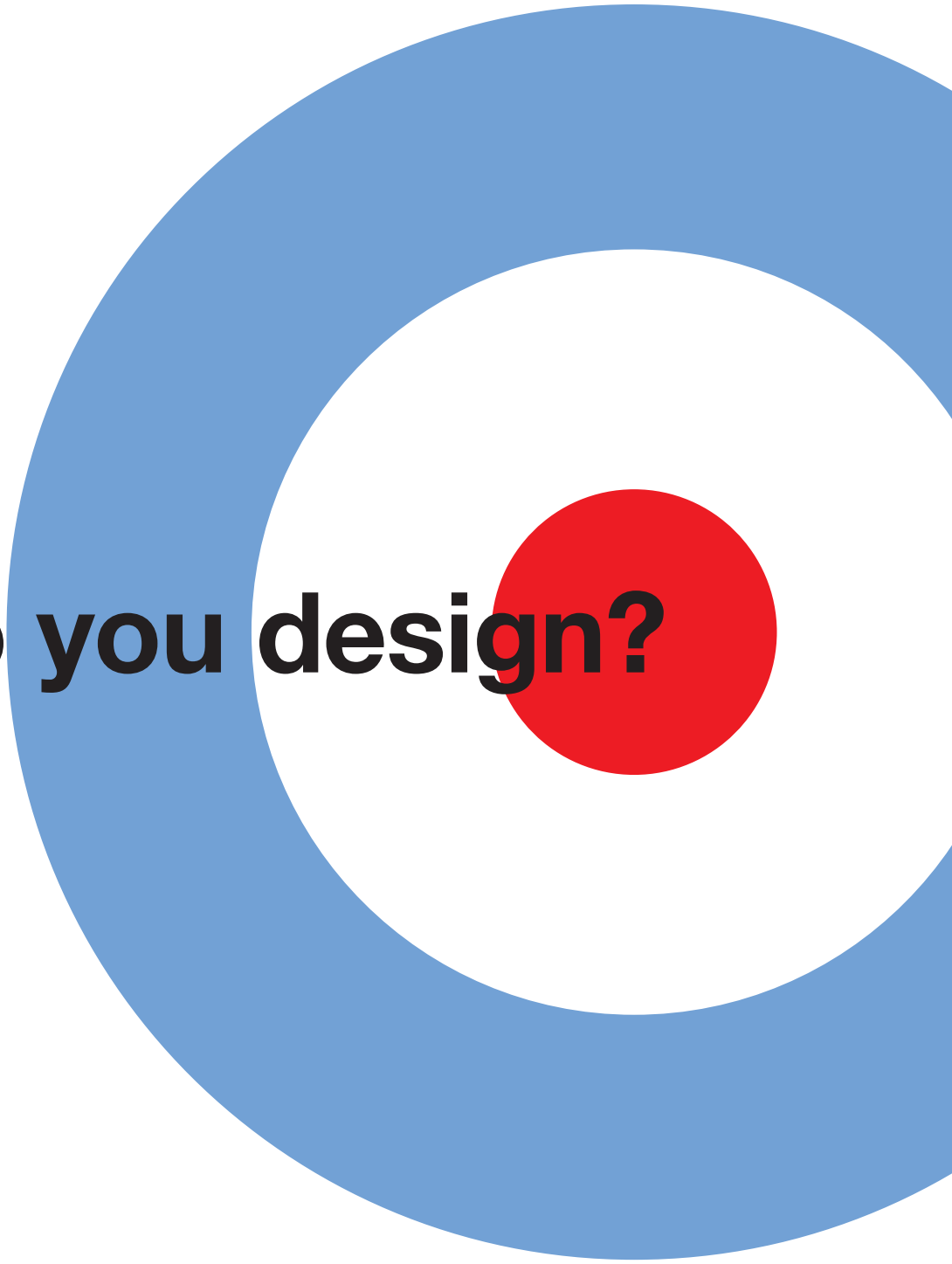


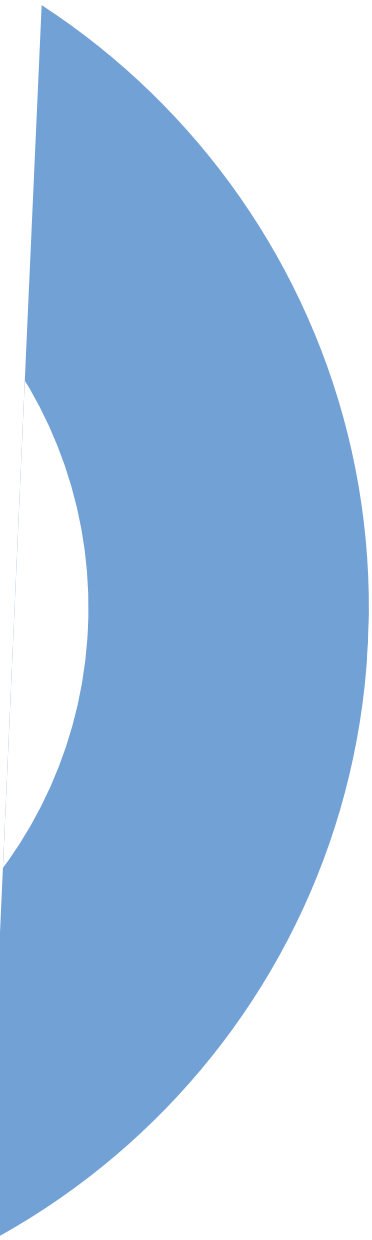
How do you design?

A Compendium of Models

by Hugh Dubberly

Dubberly Design Office
2501 Harrison Street, #7
San Francisco, CA 94110

415 648 9799



Everyone designs.

The teacher
arranging desks
for a discussion.

The entrepreneur
planning a business.

The team
building a rocket.

Their results differ.

So do their goals.
So do the scales of their projects
and the media they use.

Even their actions
appear quite different.

What's similar
is that they are designing.

What's similar
are the processes
they follow.

Our processes
determine the quality
of our products.

If we wish to improve our products,
we must improve our processes;
we must continually redesign
not just our products
but also the way we design.

That's why we study the design process.

To know what we do
and how we do it.

To understand it
and improve it.

To become better designers.

Introduction

In this book, I have collected over one-hundred descriptions of design and development processes, from architecture, industrial design, mechanical engineering, quality management, and software development. They range from short mnemonic devices, such as the 4Ds (define, design, develop, deploy), to elaborate schemes, such as Archer's 9-phase, 229-step "systematic method for designers." Some are synonyms for the same process; others represent differing approaches to design.

By presenting these examples, I hope to foster debate about design and development processes.

How do we design?
Why do we do it that way?

How do we describe what we do?
Why do we talk about it that way?

How do we do better?

Asking these questions has practical goals:
- reducing risk (increasing the probability of success)
- setting expectations (reducing uncertainty and fear)
- increasing repeatability (enabling improvement)

Examing process may not benefit everyone. For an individual designer—imagine someone working alone on a poster—focusing on process may hinder more than it helps. But teaching new designers or working with teams on large projects requires us to reflect on our process. Success depends on:
- defining roles and processes in advance
- documenting what we actually did
- identifying and fixing broken processes

Ad hoc development processes are not efficient and not repeatable. They constantly must be reinvented making improvement nearly impossible. At a small scale, the costs may not matter, but large organizations cannot sustain them.

From this discussion, more subtle questions also arise:

How do we minimize risk while also maximizing creativity?

When must we use a heavy-weight process?
And when will a light-weight process suffice?

What is the place of interaction design within the larger software development process?

What is the place of the software development process within the larger business formation processes?

What does it mean to conceive of business formation as a design process?

Origins

The oldest development process model I've seen dates from about 1920 and describes how to develop a battleship for the Royal Navy. Discussions about design and development processes began in earnest shortly after the second world war. They grew out of military research and development efforts in at least three fields, operations research, cybernetics, and large-scale engineering project management.

Pre-war efforts to make radar an effective part of the British air-defense system led to operations research, which then matured into an academic discipline. Development of automatic piloting devices and fire-control systems for aiming large guns led to servo-mechanisms and computing devices, anticipating the emergence of cybernetics, one of the roots of artificial intelligence. Large engineering projects undertaken during the war and later cold-war projects, such as the Atlas and Titan missile projects, demanded new techniques to deal with increased scale and complexity.

The excitement of these new disciplines and the success of these huge engineering projects captivated many people. From operations research, cybernetics, and large-scale engineering project management, academic designers imported both methods and philosophy in what became known as the design methods movement (1962-1972). Work in the UK, at Ulm in Germany, and MIT and Berkeley in the US sought to rationalize and systemize the design process. Several designers attempted to codify the design process and present it as a scientific method.

Somewhat parallel efforts occurred in the business world. Stafford Beer and others applied systems thinking and especially operations research to business problems. During the 1950s, W. Edward Deming examined business processes. His work led to the quality management movement, which became popular in Japan and something of a fad in the US in the 1980s. Its principles became standard operating procedures in much of the business world, becoming enshrined in ISO and six-sigma standards.

In the software world, interest in the development process dates back at least to the IBM System 360, released in 1964. In 1975, Fred Brooks, manager of OS/360, published *The Mythical Man Month,* his "belated answer to [IBM Chairman] Tom Watson's probing question as to why programming is so hard to manage."

Today, software developers are still actively discussing the question. Consultants seek to differentiate themselves with proprietary processes. Software tools makers seek standards around which they can build tools—a new twist on codifying the design process.

Curious ties exists between the design methods world and the software development world. One of the founders of the design methods movement, Christopher Alexander, co-wrote *A Pattern Language.* Alexander's work on design patterns in architecture contributed to thinking on design patterns in software. In the 1970s, another important figure in the movement, Horst Rittel, developed IBIS (Issues-Based Information System) to support the design process. Rittel's research into IBIS is a precursor of today's work on design rationale.

But for the most part, designers, business managers, and software developers appear to be unaware of practices and thinking about process in the other disciplines. Even within their own fields, many are unaware of much prior art.

The fields overlap, but so far as I know, no one has attempted to bring together work from these three areas. One of my goals is to cast each of these activities as design, to show how their processes are similar, and to encourage sharing of ideas between the disciplines.

Measure twice
Cut once

Ready
Aim
Fire

Lab study
Pilot plant
Full-scale plant

Research
Development
Manufacturing
Sales

Contents

The carpenter's adage,
the captain's command,
the chemical engineer's "scale-up" process,
the corporation's departments—
the four phrases on the previous page—
have something in common.

Each is a sequence of steps.

Each is a process focused on achieving a goal.

Each suggests iteration and convergence.

Each is an analog of the design process.

This book presents many other descriptions of design and development processes. I call these descriptions design process models, distinguishing the description from the activity it describes. I also combine design and development into one category, because the distinction is without a difference.

This collection is not exhaustive. Even so, organizing it presents a challenge. At the end of the book are indices organized by title, date, and author. In the body of the book are threads but no strong narrative. I have paired models where I see a connection. These pairings—and the entire structure—are idiosyncratic. Thus, the book is more a reference work than a primer.

11 Introducing process

19 Analysis versus synthesis

29 Academic models

61 Consultant models

67 Software development

82 Complex linear models

115 Cyclic models

132 Complete list of models

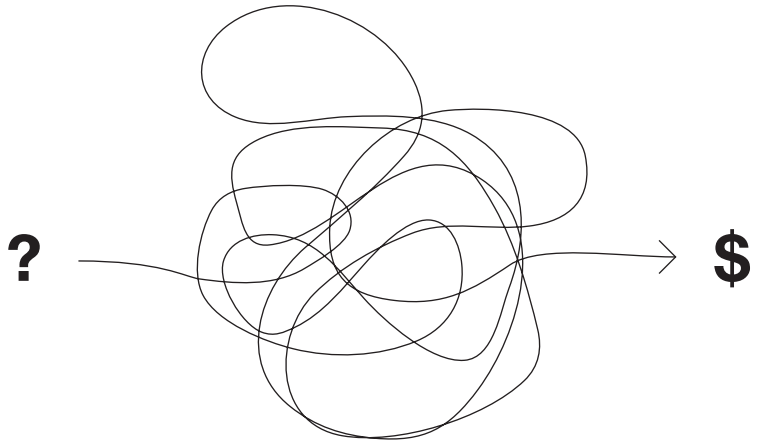
136 Chronological list

140 Author list

Design process
after Tim Brennan (~1990)

At an off-site for Apple Computer's Creative Services department, Tim Brennan began a presentation of his group's work by showing this model. "Here's how we work," he said. "Somebody calls up with a project; we do some stuff; and the money follows."

Brennan captures important aspects of the process:
- the potential for play
- its similarity to a "random walk"
- the importance of iteration
- its irreducible "black-box" nature



Introducing process

What is a process?
Where does it begin?
Where does it end?
How much detail is enough?

We begin with simple models
of the design process
and look at how they might be expanded
into useful frameworks.

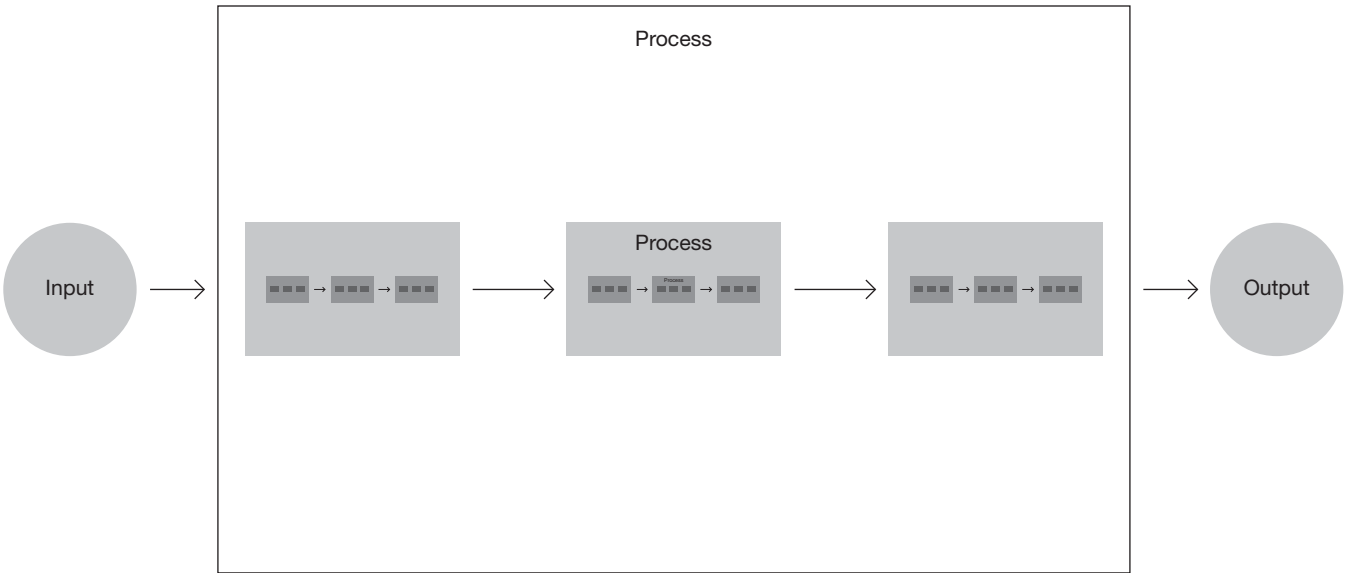
Process archetype

A process must have input and output. Garbage in; garbage out. (Good in; good out?) In between, something may happen—the process—a transformation. Sometimes, the transformation is reducible to a mathematical function. Think of using Photoshop's curves function to lighten a photo. One risk in using this framework is that it neatens a messy world. It may promote an illusion of linearity and mechanism—of cause and effect.

On the infinite expandability of process models

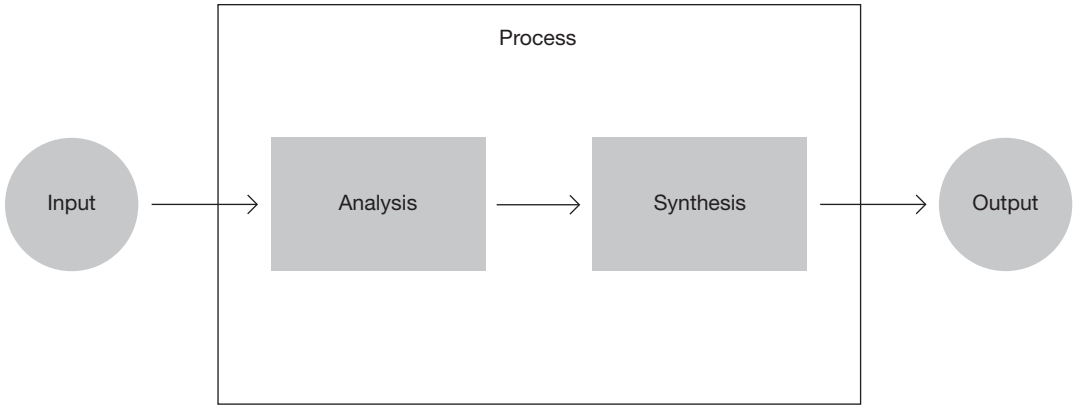
An important step in managing any process is documenting it. That truism implies a process merely needs recording. But documenting a process is like taking a photograph. The author chooses where to point the camera—where to begin mapping the process, where to end, what to put in, what to leave out, how much detail to include.

Processes have a fractal quality. You can zoom in or out, increasing or decreasing abstraction or specificity. You can add more detail—dividing phases into steps and steps into sub-steps, almost infinitely. Processes rarely have fixed beginnings or endings. You can almost always add steps upstream or downstream.



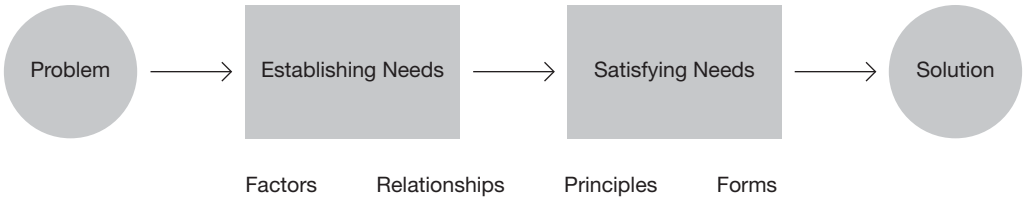
Design process archetype: Analysis, Synthesis
after Koberg and Bagnall (1972)

"When comparing many different problem-solving approaches it becomes necessary to search for their basic abstractions or common-denominators," write Koberg and Bagnall. "If you'll try it yourself, we're sure that the two "basic" stages of *analysis* and *synthesis* will emerge; i.e., when consciously solving problems or when creatively involved in the activity of design, two basic stages are necessary. *First*, we break the situation or whole problem into parts for examination (Analysis) and *Second*, we reassemble the situation based on our understanding of improvements discovered in our study (Synthesis)."



Problem, Solution
after JJ Foreman (1967)

Foreman, like Koberg and Bagnall, casts design as problem-solving. This stance is typical of the first generation of the design methods movement. Foreman introduces the idea of needs. He also begins to sub-divide the process.

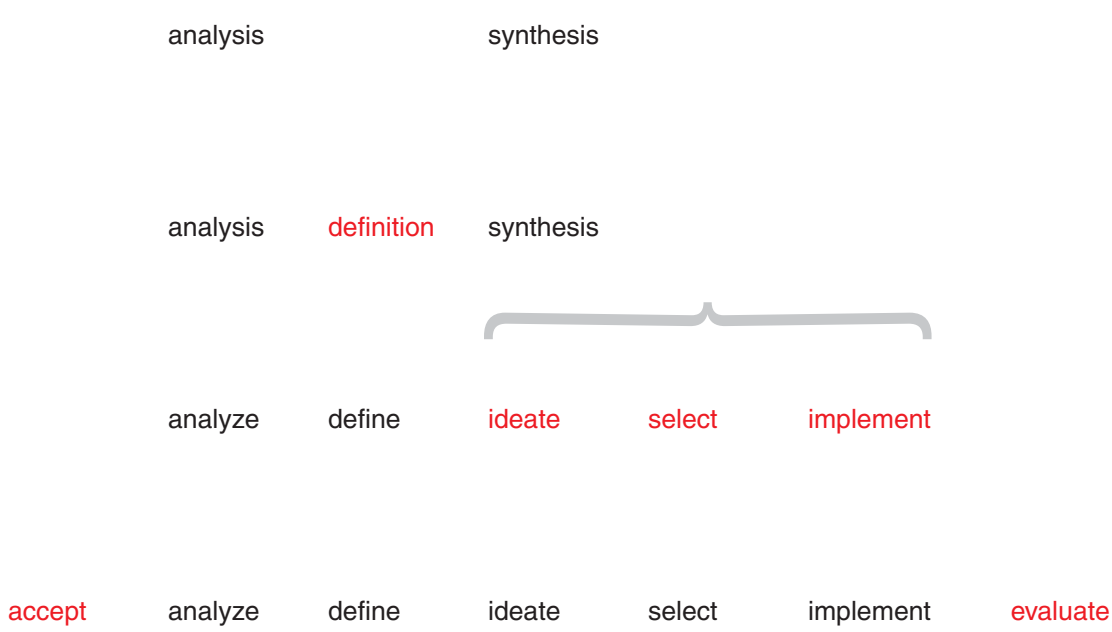


Expanding the two-step process
after Don Koberg and Jim Bagnall (1972)

In their classic book, *The Universal Traveler*, Koberg and Bagnall (who taught in the College of Environmental Design at Cal Poly in San Luis Obisbo) expand the archtypal two-step process to three, then five, and finally to seven steps.

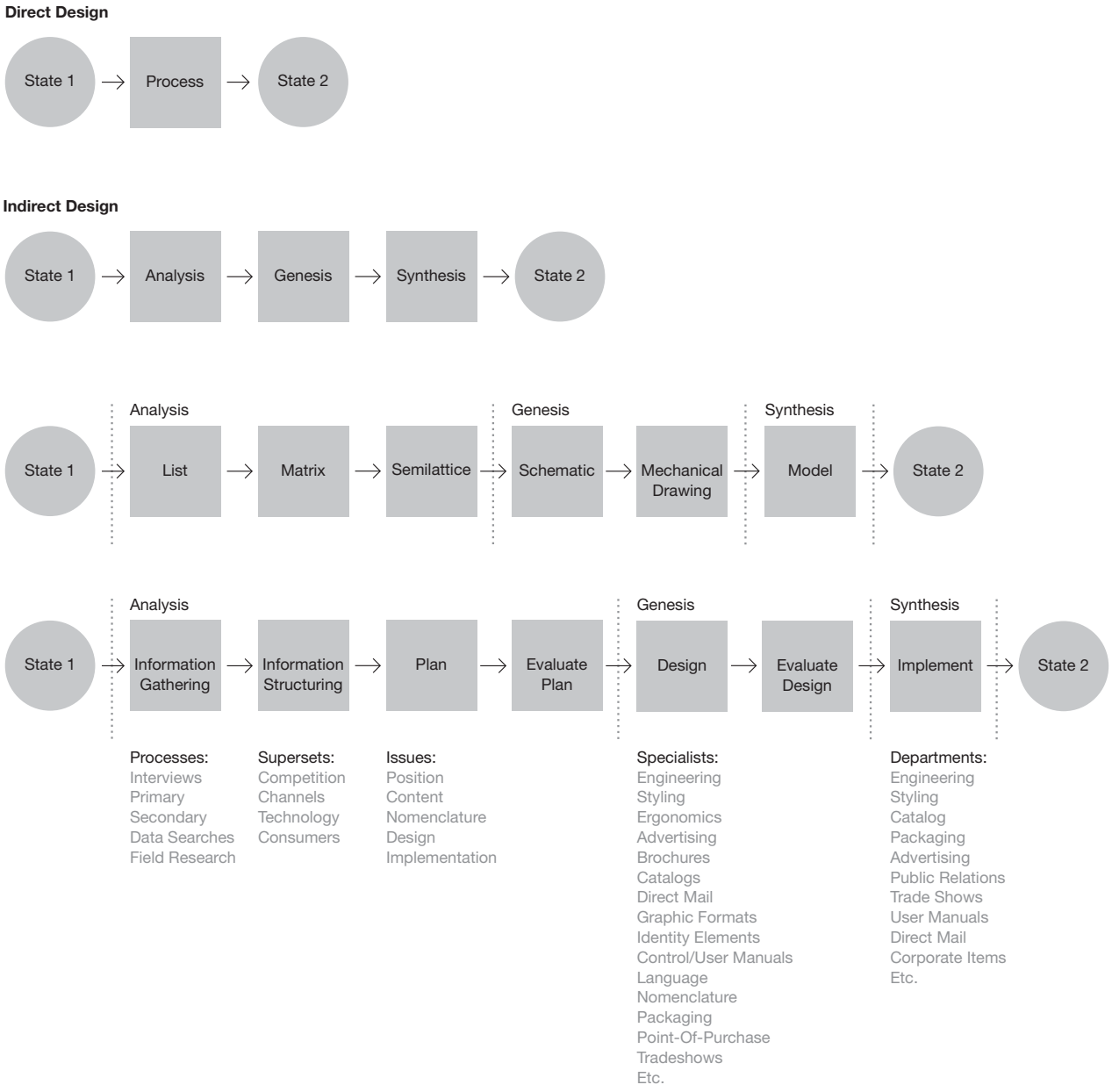
They note “that ‘out of Analysis’ we derive an understanding or concept that is then followed as a guideline in the rebuilding or Synthesis stage.” Within the book’s “problem-solving” frame, definition becomes problem definition, and they never follow up on the idea of definition as concept or parti.

The synthesis phase becomes “ideate, select, implement,” while the analysis phase remains intact. Finally, they add a new phase at the beginning and another at the end.



Matching process to project complexity
after Jay Doblin (1987)

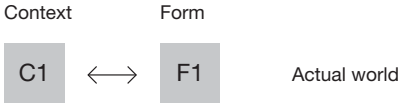
In his article, “A Short, Grandiose Theory of Design,” Doblin presents a similar series of expanding processes. Dobin’s notion of direct and indirect design echos Alexander’s (1962) model of unselfconscious and self-conscious design. Doblin’s third and fourth processes correspond to Alexander’s third type of design, mediated design (my title). (For more on Alexander’s model, see the next page.)



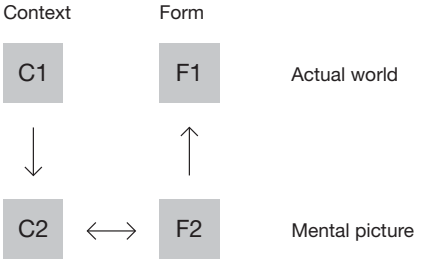
Unself-conscious and self-conscious design
after Christopher Alexander (1962)

In *Notes on the Synthesis of Form*, Alexander (1962) described three situations in which designing may take place. In the first, a craftsman works directly and unself-consciously through "a complex two-directional interaction between the context C1 and the form F1, in the world itself." In the second, designing is separate from making. Form is shaped "by a conceptual picture of the context which the designer has learned and invented, on the one hand, and ideas and diagrams and drawings which stand for forms, on the other." In the third, the designer also works self-consciously, this time abstracting and formalizing representations of the problem and solution so that he and others may inspect and modify them.

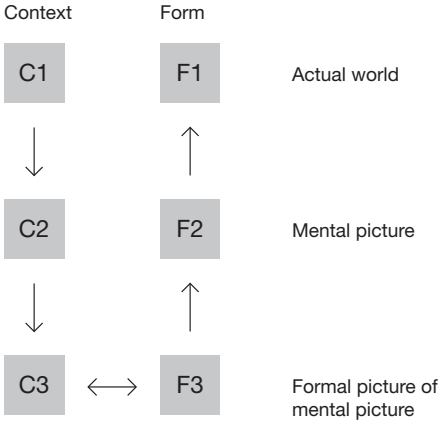
1. Un self conscious



2. Self conscious



3. Mediated



Analysis
synthesis
evaluation

In 1962, Jones proposed this procedure as a basic framework for design processes.

But what relationship do the steps have?
Are they discrete?
Sequential?
Overlapping?

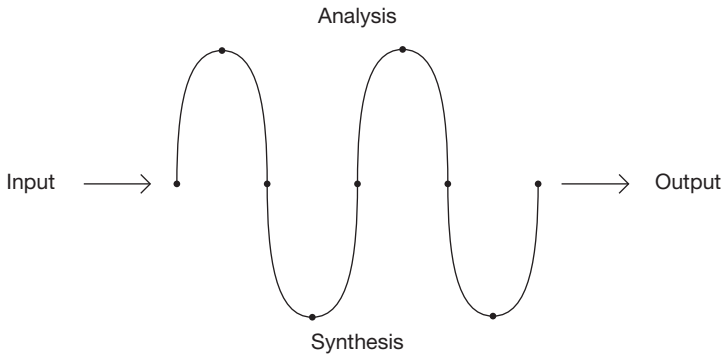
This section compares several models.

While attention often focuses on the analysis-synthesis dichotomy, we might also consider other dichotomies:

serialist versus holist
linear versus lateral
top-down versus bottom-up
agile versus heavy-weight
pliant versus rigid

Oscillation

We may view the design process as an oscillation of the designer's attention between analysis and synthesis. Do wave-length and amplitude remain constant? Do they vary over time? What are the beginning and ending conditions?

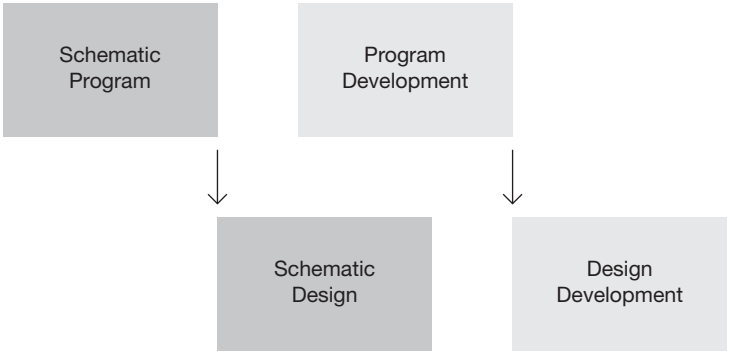


Programming and designing
after William M. Pena and Steven A. Parshall (1969)

This model comes from architecture, where programming refers not to computers but to a phase of planning that precedes design of a building. Pena and Parshall quote Webster, "[Programming is] a process leading to the statement of an architectural problem and the requirements to be met in offering a solution." They describe programming as "problem seeking" and design as "problem solving."

They note, "Programming IS analysis. Design IS synthesis."

Pena and Parshall recommend "a distinct separation of programming and design." "The separation of the two is imperative and prevents trial-and-error design alternatives."



Programming concerns five steps:

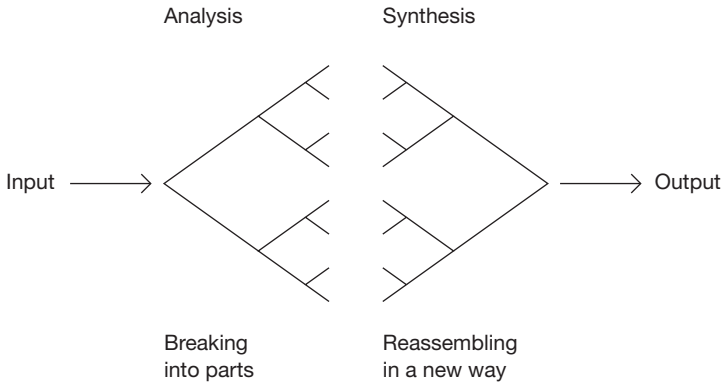
1 Establish Goals
What does the client want to achieve, and Why?
2 Collect and analyze Facts
What do we know? What is given?
3 Uncover and test Concepts
How does the client want to achieve the goals?
4 Determine Needs
How much money and space? What level of quality?
5 State the Problem
What are the significant conditions affecting the design of the building?
What are the general directions the design should take?

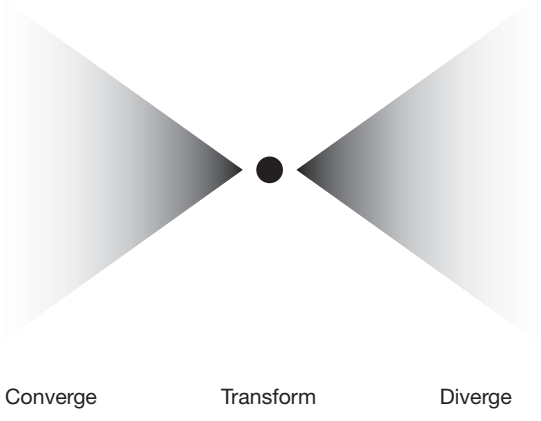
Diverge / Converge vs Narrow / Expand

Often designers describe themselves as creating many options (diverging) and then narrowing down their options (converging). Alexander (1962) and other designers have described analysis as a process of breaking a problem into pieces—of "decomposing" it. Synthesis follows as re-ordering the pieces based on dependencies, solving each sub-piece, and finally knitting all the pieces back together—"recombining" the pieces. This decomposition-recombination process also diverges and then converges.

We may just as easily describe the process by reversing the sequence (narrowing down, expanding out). Analyzing a problem leads to agreement—to definition—a convergent process. At that point, hopefully, the "miracle" of transformation occurs in which the solution concept arises. Then, the designer elaborates that concept in greater and greater detail—a divergent process.

Later, we see this question arise again in the section on spiral models. Some (Souza) converge on a solution. Others (Boehm) diverge from a center, suggesting the accumulation of detail. (See pages 122-125.)

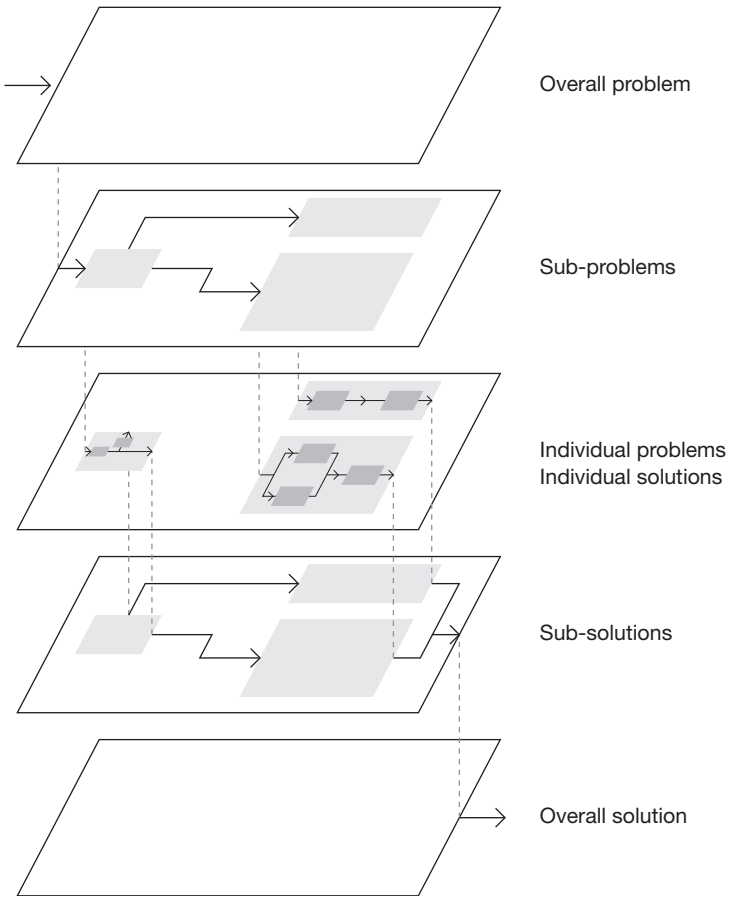




Decomposition / recombination
after VDI 2221 (from Cross 1990)

VDI 2221 mirrors Alexander's decomposition-recombination process. Cross wrote, "The VDI Guideline follows a general systemic procedure of first analyzing and understanding the problem as fully as possible, then breaking this into sub-problems, finding suitable sub-solutions and combining these into an overall solution."

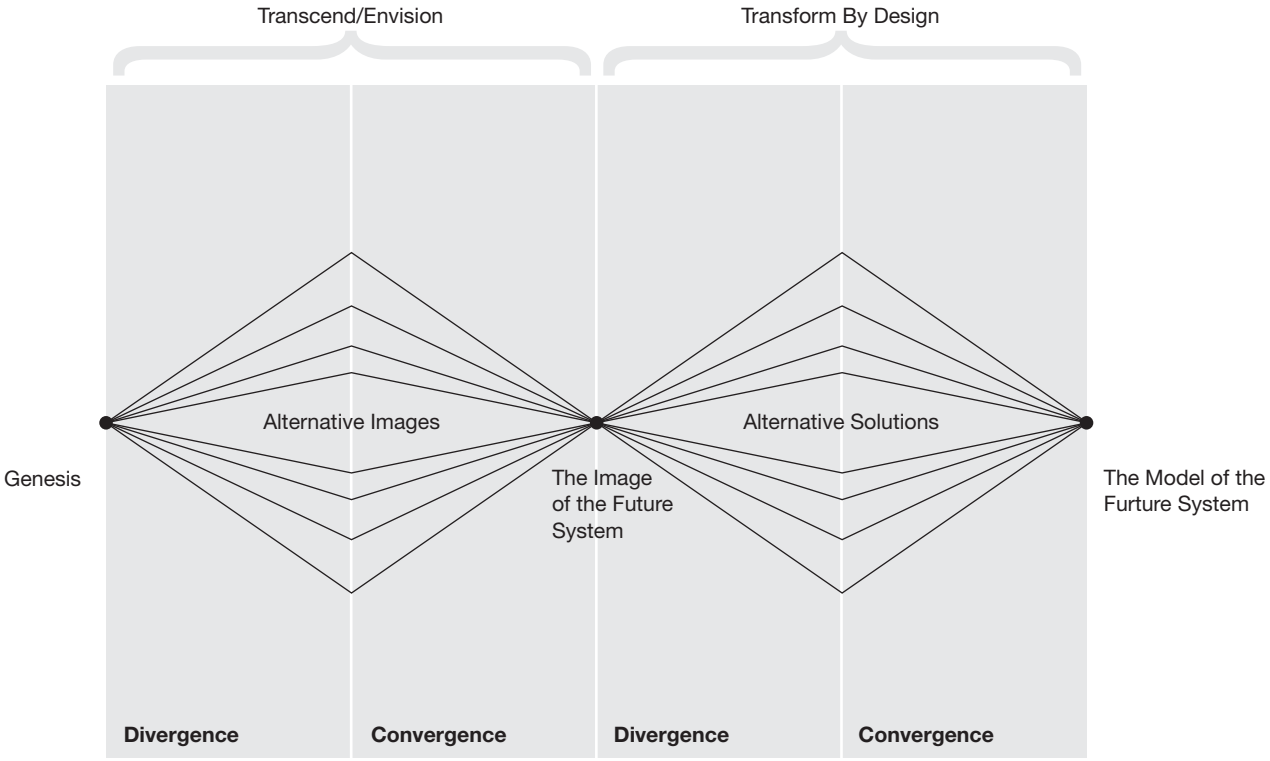
"This kind of procedure has been criticized in the design world because it seems to be based on a problem-focused, rather than a solution-focused approach. It therefore runs counter to the designer's traditional ways of thinking." (For another view of VDI 2221, see page 32.)



Dynamics of divergence and convergence
after Bela H. Banathy (1996)

Banathy's model illustrates the iterative nature of the design process, repeating the process of divergence and convergence, analysis and sysnthesis.

In Banathy's view, "We first diverge as we consider a number of inquiry boundaries, a number of major design options, and sets of core values and core ideas. Then we converge, as we make choices and create an image of the future system. The same type of divergence-convergence operates in the design solution space. For each of the substantive design domains (core definition, specifications, functions, enabling systems, systemic environment) we first diverge as we create a number of alternatives for each, and then converge as we evaluate the alternatives and select the most promising and most desirable alternative."

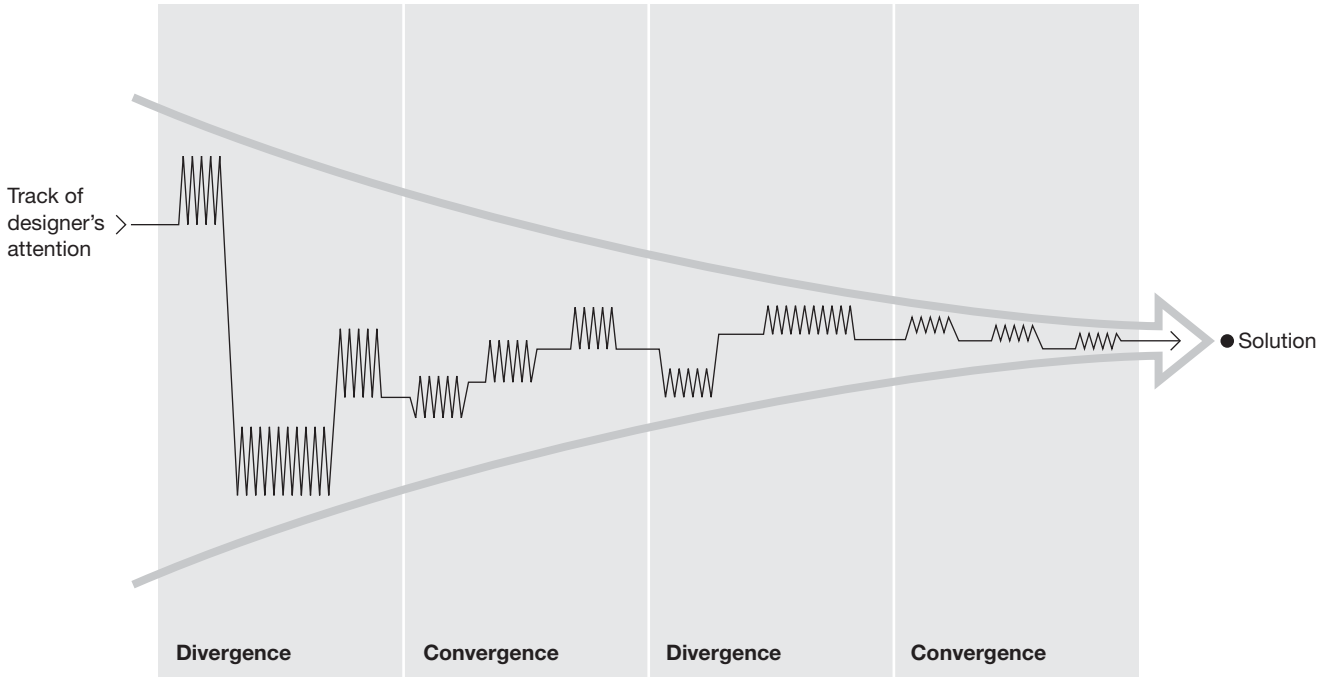


Overall, the design process must converge
after Nigel Cross (2000)

Cross notes, “Normally, the overall aim of a design strategy will be to converge on a final, evaluated and detailed design proposal, but within the process of reaching that final design there will be times when it will be appropriate and necessary to diverge, to widen the search or to seek new ideas and starting points.

The overall process is therefore convergent, but it will contain periods of deliberate divergence.”

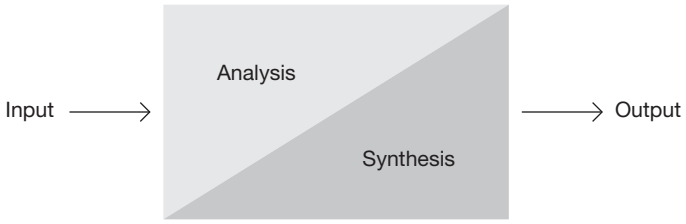
Banathy’s and Cross’s models suggest cycles and are similar to the iterative process of Marcus and Maver (see page 45) and to the spirals of Boehm and others (see pages 122-125).



Gradual shift of focus from analysis to synthesis
after Bill Newkirk (1981)

Bill Newkirk first taught me that synthesis begins at the very beginning of a design project. Koberg and Bagnall (1972) suggested that both analysis and synthesis continue throughout a project. Designers may begin by focusing on analysis and gradually shift their focus to synthesis.

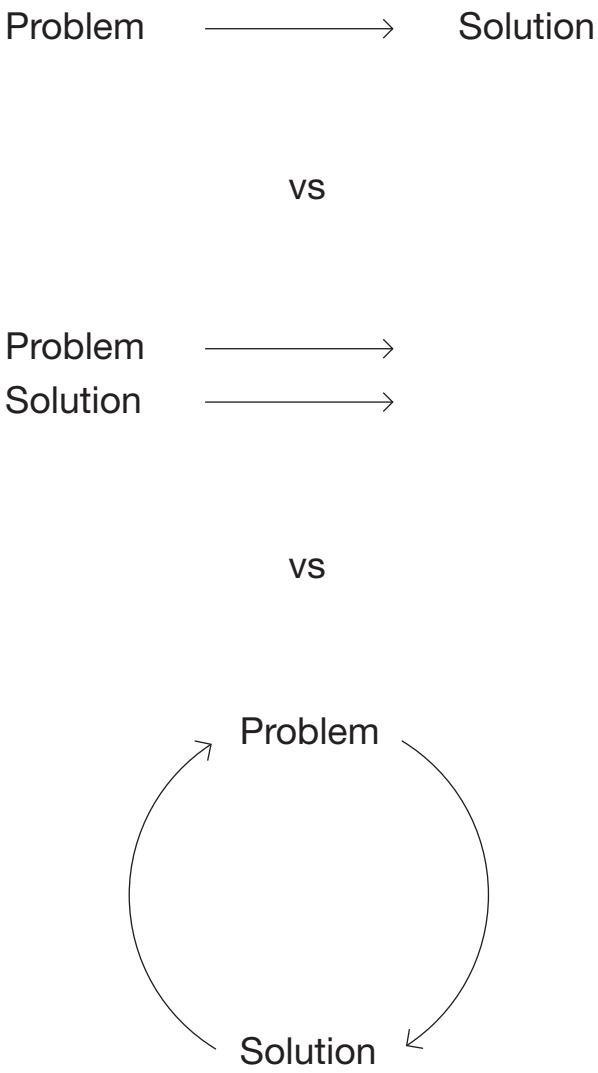
Lawson (1990) notes, “Most of the maps of the design process which we have looked at seem to resemble more closely the non-designer, scientist approach than that of the architects: first analysis then synthesis. For the designers it seems, analysis, or understanding the problem is much more integrated with synthesis, or generating a solution.” He reports studies by Eastman (1970) and Akin (1986) confirming this view. “Akin actually found that his designers were constantly both generating new goals and redefining constraints. Thus, for Akin, analysis is a part of all phases of design and synthesis is found very early in the process.”



Problem to solution: sequence, parallel process or loop?

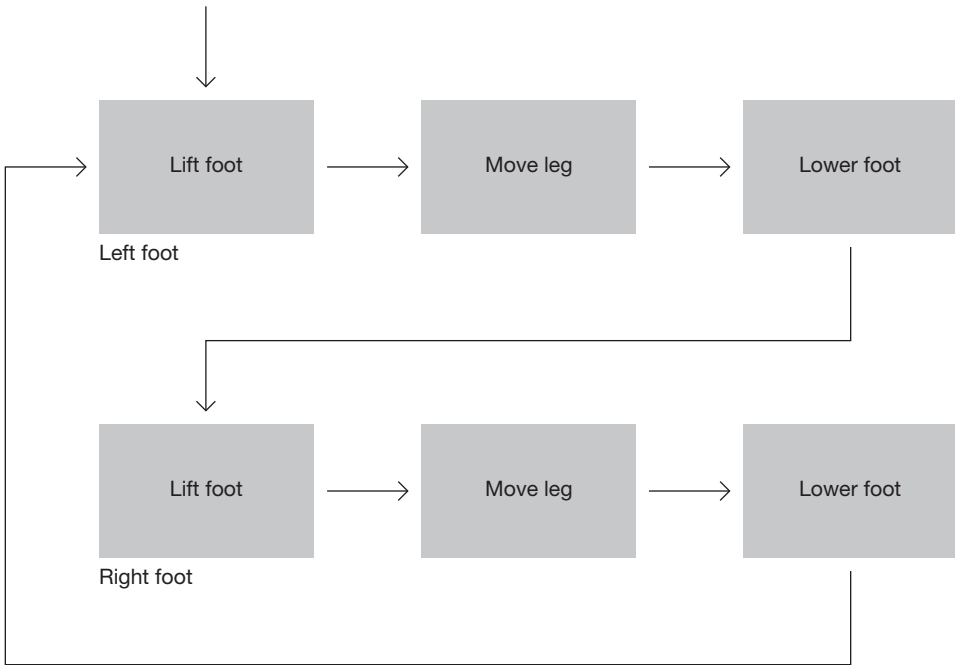
Pena and Parshall (1969), Briggs and Havlick (1976), and others, particularly in the early phases of the design methods movement, described problem solving as a sequential activity. In this model, we must define a problem before we can solve it.

On the other hand, most people agree that a solution is inherent in a problem. Having defined a problem, we've defined or at least outlined the solution. Rittel and Webber (1973) note, "The information needed to *understand* the problem depends upon one's idea for *solving* it." (Italics are theirs.) "Problem understanding and problem resolution are concomitant to each other." Attempting to solve a problem (prototyping) may even improve our understanding of a problem—and thus change our definition.



Walking process
after Lawson (1980)

Bryan Lawson offered this map “with apologies to those design methodologists who like maps!” He notes that many models of the design process are “theoretical and prescriptive” rather than descriptions of actual behavior.



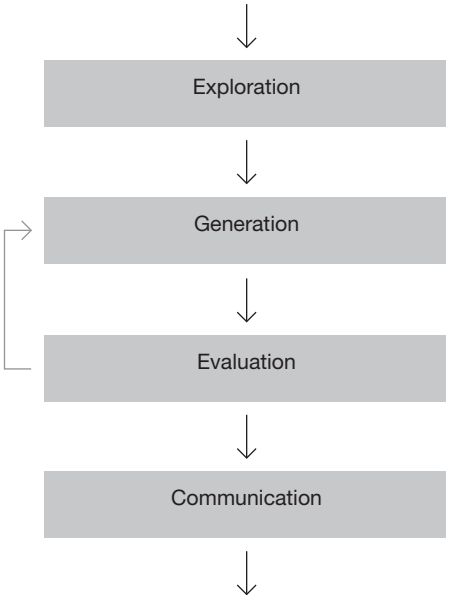
Academic models

Many teachers in the design fields, engineering, and architecture have developed models of the design process to help their students learn to design.

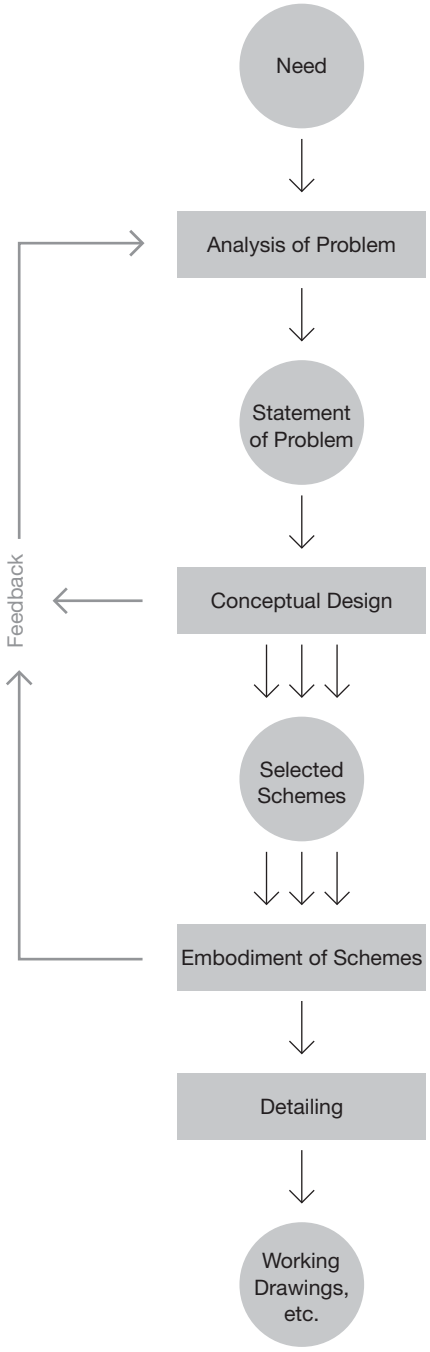
Four stage design process
after Nigel Cross (2000)

Writing from an engineering perspective, Cross developed this “simple descriptive model of the design process, based on the essential activities that the designer performs. The end-point of the process is the communication of a design, ready for manufacture. Prior to this, the design proposal is subject to evaluation against the goals, constraints and criteria of the design brief. The proposal itself arises from the generation of a concept by the designer, usually after some initial exploration of the ill-defined problem space.”

Cross’s model includes communication as a final stage. Archer (1963) may have been the first to include communication as an explicit stage in a design process model. (See page 98.)



Engineering design process
after Michael J. French (1985)



French also wrote from an engineering perspective. He suggested, “The analysis of the problem is a small but important part of the overall process. The output is a statement of the problem, and this can have three elements:
- a statement of the design problem proper
- limitations placed up the solution,
  e.g. codes of practice, statutory requirements, customers’ standards, date of completions
- the criterion of excellence to be worked to.”

The conceptual design phase “takes the statement of the problem and generates broad solutions to it in the form of schemes. It is the phase that makes the greatest demands on the designer, and where there is the most scope for striking improvements. It is the phase where engineering science, practical knowledge, production methods and commercial aspects need to be brought together . . .”

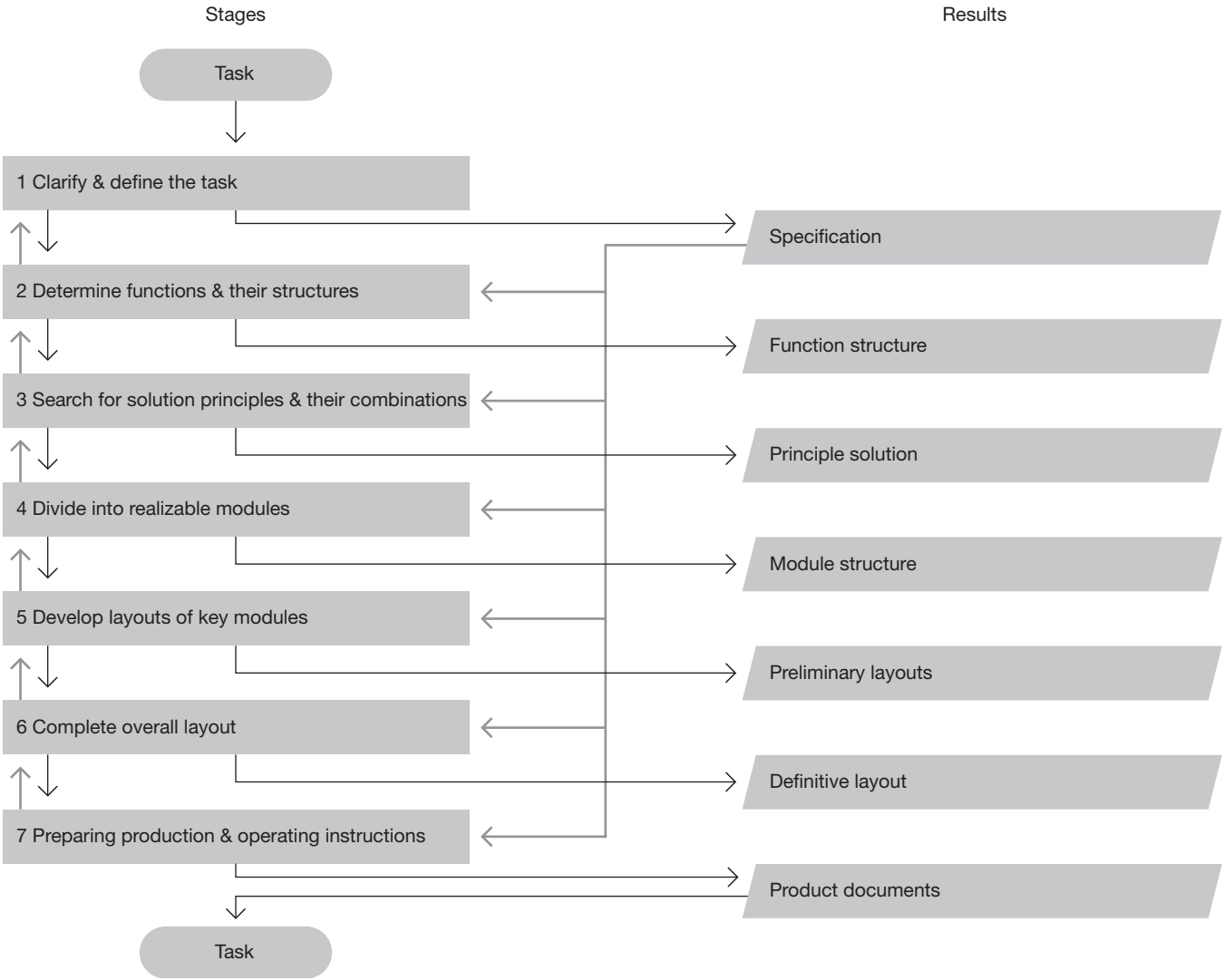
In the third phase, “schemes are worked up in greater detail and, if there is more than one, a final choice between them is made. The end product is usually a set of general arrangement drawings. There is (or should be) a great deal of feedback from this phase to the conceptual design phase.

In the detailing phase, “a very large number of small but essential points remain to be decided.”

System approach to the design of technical systems and products
after Verein Deutscher Ingenieure (1987)

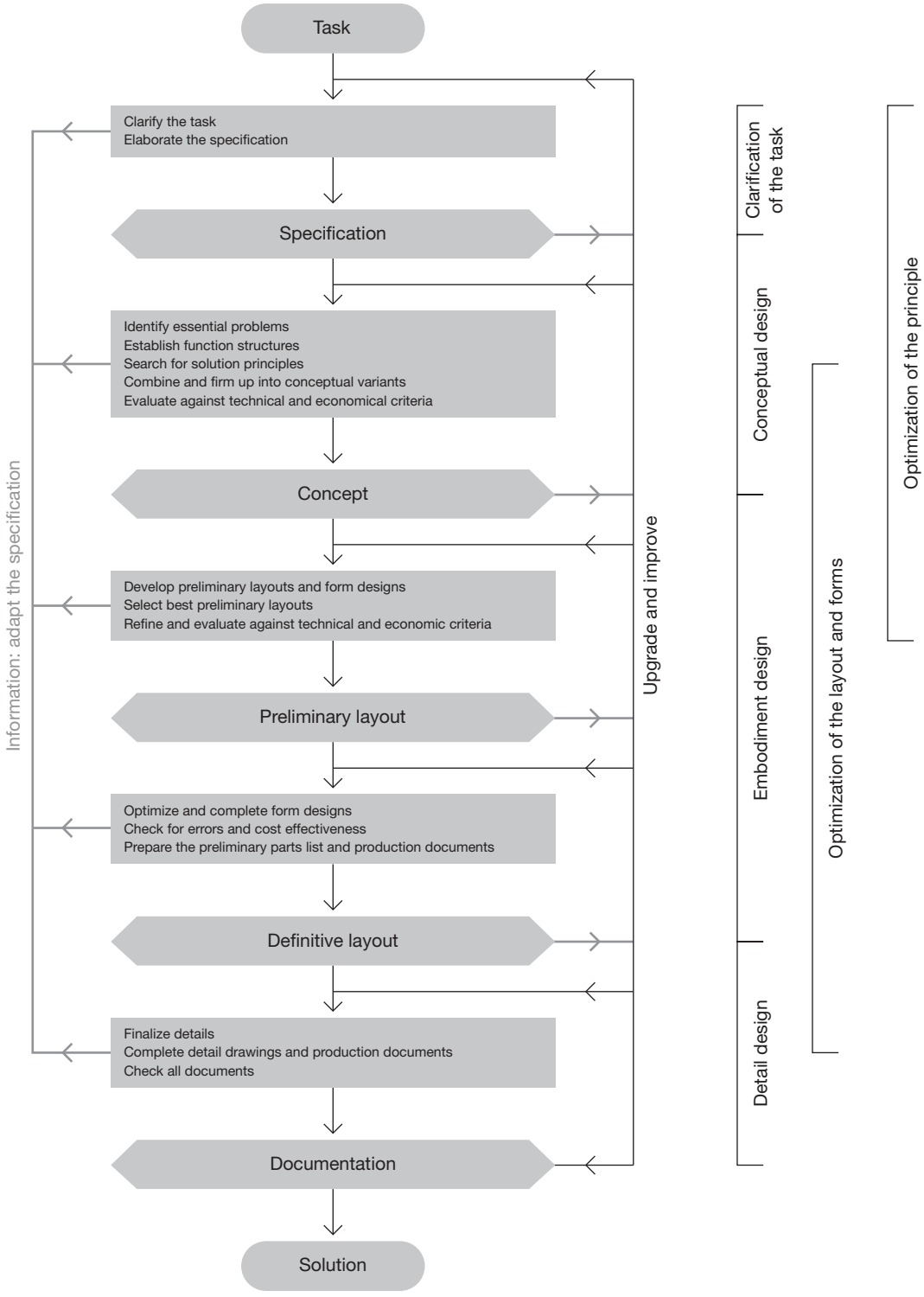
VDI stands for Verein Deutscher Ingenieure, the professional engineering society of Germany. Their guideline 2221 suggests, “The design process, as part of product creation, is subdivided into general working stages, making the design approach transparent, rational and independent of a specific branch of industry.”

The full process contains much more detail than the diagram below shows. In practice, the process is less linear than the diagram implies. “It is important to note that the stages do not necessarily follow rigidly one after the other. They are often carried out iteratively, returning to preceding ones, thus achieving a step-by-step optimization.”



Design process
after Gerhard Pahl and Wolfgang Beitz (1984)

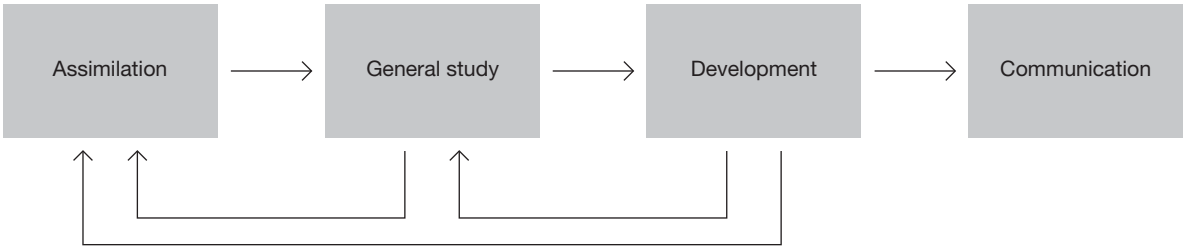
Cross recommends this model as “reasonably comprehensive” but not obscuring “the general structure of the design process by swamping it in the fine detail of the numerous tasks and activities that are necessary in all practical design work.” He seems to refer to Archer’s “Systematic method for designers”. (See page 98.)



Architect's plan of work (schematic)
after the RIBA Handbook (1965)

Lawson presents this model from the RIBA (Royal Institute of British Architects) practice and management handbook. According to the handbook, assimilation is "The accumulation and ordering of general information specifically related to the problem in hand." General study is "The investigation of the nature of the problem. The investigation of possible solutions or means of solution." Development is "refinement of one or more of the tentative solutions isolated during phase 2." Communication involves describing "one or more potential solutions to people inside or outside the design team."

Lawson is critical, "it is hardly a map at all. . . . In short, all this map does is to tell us that designers have to gather information about a problem, study it, devise a solution and draw it, though not necessarily in that order."



Architect's plan of work, (detailed)
after the RIBA Handbook (1965)

The handbook also contains another, more detailed plan of work occupying 27 pages. The 12 main stages are described below. Lawson criticizes this model as "a description not of the process but of the products of that process. . . . It's also worth noting that the stages in the Plan of Work are closely related to the stages of fee payment in the Conditions of Engagement for Architects. So the Plan of Work may also seen as part of a business transaction; it tells the client what he will get, and the architect what he must do rather than how it is done. In the detailed description of each section the Plan of Work also describes what each member of the design team (quantity surveyor, engineers etc) will do, and how he will relate to the architect; with the architect clearly portrayed as the manager and leader of this team. This further reveals the Plan of Work to be part of the architectural profession's propaganda exercise to stake a claim as leader of the multi-disciplinary building design team."

Briefing	Inception Feasibility
Sketch plans	Outline proposals Scheme design
Working drawings	Detail design Production information Bills of quantities Tender action
Site operations	Project planning Operations on site Completion Feed-back

Problem solving process
after George Polya (1945)

In 1945, George Polya wrote *How to Solve It*, an excellent little book for students and teachers of mathematics. In it, he describes a process for solving math problems, though one might apply his process more generally.

Many in the design methods movement seem to have been familiar with Polya's book. Bruce Archer (1963-1964) mentions Polya in his booklet, *Systemic method for designers*. Likewise, Maldonado and Bonsiepe (1964) also mention Polya in their article "Science and Design."

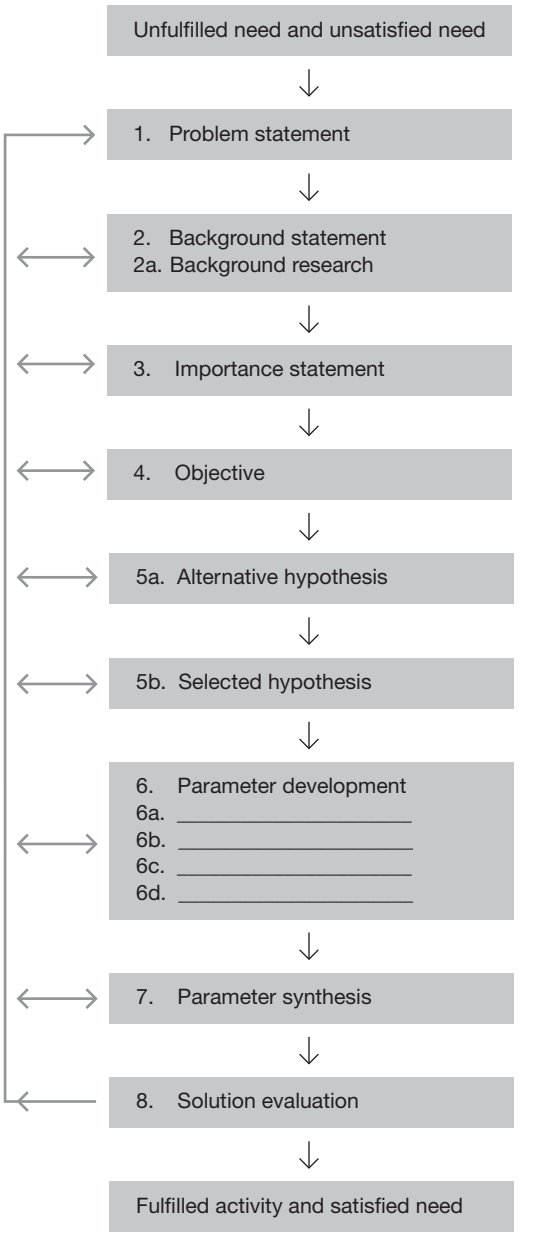
Thus Polya seems to have influenced the teaching of architecture, as may be seen in the "scientific problem solving process" described on the following page.

1. Understanding the problem	What is the unknown? What are the data? What is the condition? Draw a figure. Introduce suitable notation. Separate the various parts of the condition. Can you write them down?
2. Devising a plan	Find the connection between the data and the unknown. Do you know a related problem? Look at the unknown! Here is a problem related to yours and solved before. Could you use it? Could you restate the problem? Could you restate it still differently? Go back to definitions. You should obtain eventually a plan of the solution.
3. Carrying out the plan	Check each step. Can you see clearly that the step is correct? Can you prove that it is correct?
4. Looking back	Check the result. Can you derive the result differently? Can you use the result, or the method, for some other problem?

Scientific problem solving process
after Cal Briggs and Spencer W. Havlick (1976)

Briggs and Havlick used this model for teaching design to undergraduates at the University of Colorado's College of Environmental Design. The college's name implies links to environmental design faculties at Berkeley, San Luis Obispo, and Ulm and thus to the design methods movement. Briggs and Havlick shared the early movement's desire to cast design as a science.

They write, "the role of the environmental designer is to solve human environmental problems by the creation and implementation of optimal physical form. . . . The scientific method is the central process. [We have] borrowed the scientific method from the traditional sciences and adapted it for the development of optimal solutions. Termed the scientific problem solving design process, it has been utilized to insure an analytic, systematic, and precise approach to the solution of man's environmental malfunctions."



1. The delineation of a malfunction in the human environment which was created by the inability to perform a needed human activity.

2. The investigation into background research which answers the who, what, where, how, when, and why of the problem. This stage also explores
a) previously attempted physical form solutions and
b) all contributing causes to the problem.

3. A statement which demonstrates the significance of the environmental malfunction in terms of critical human needs and what consequences might result if the problem was not solved.

4. A statement of the intended goals that will be fulfilled by the solution of the problem.

5. A researched assumption which asserts that if a certain physical form approach is taken, certain results will ensue (the satisfaction of unmet need).

6. The definition and ordering of all design considerations necessary for the development of an optimal solution to a stated problem (i.e., legality, economic feasibility, ecological implications, shape, color, weight, etc.). Parameters or variables are the form determining components which comprise the proposed hypothesis.

7. The systematic optimizing and compromising of ordered design parameters into a resultant physical form solution which optimally achieves the stated objectives. It should be stated here the "physical form" could be the writing of environmental legislation, redesign or creation of a product, or the formulation of how a service could be best provided.

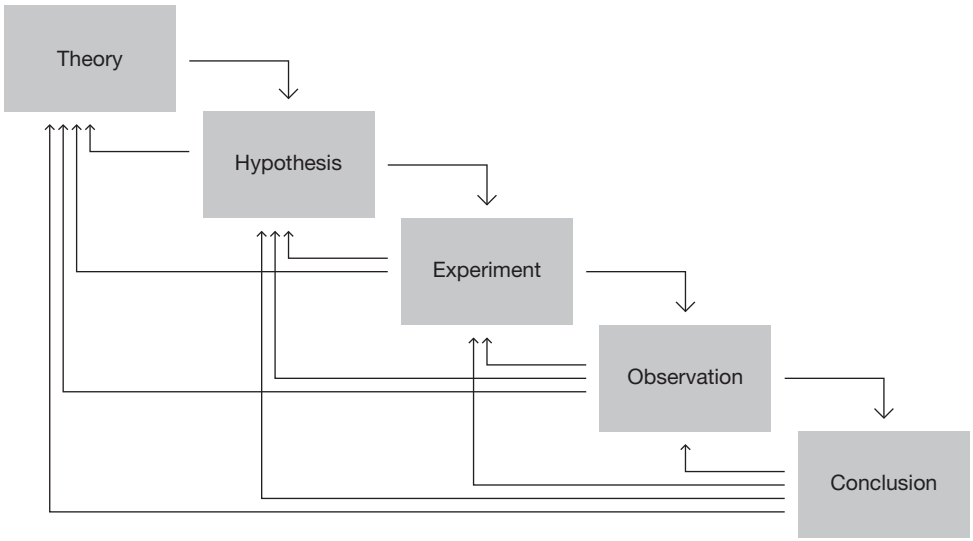
8. The implementation and testing of the generated design solution to delineate the need for any further modifications or recycling back through the scientific problem solving process. This evaluation process precedes implementation of a solution.

The arrows in the flowchart reveal an important dimension of the problem solving design process. At every procedural stage, the Environmental Designer is able to recycle back through the process one or more stages if modifications are necessary to improve the evolution of the final design.

THEOC, a model of the scientific method

THEOC is an acronym for theory, hypothesis, experiment, observation, conclusion — an easy way to remember an outline of the scientific method. It approximates the process with these steps:

- within a framework of a Theory
- generate a Hypothesis about a phenomenon
- run an Experiment to test the hypothesis
- Observe and record the results
- form a conclusion based on the relation of the observations to the hypothesis.
- repeat as necessary



Criteria of Validation of Scientific Explanations (CVSE)
after Humberto Maturana (1987)

Claudia L'Amoreaux contributed the models below comparing Maturana's view of scientific explanation with his view of the scientific method. L'Amoreaux points out that "Maturana shows you not only don't need objectivity to do science, you can't be objective. While the traditional pose of scientific objectivity may be fine in some areas, we cannot understand perception and the nervous system within that framework." Nor can we understand design that way.

Maturana writes, "scientific explanations are not valid in themselves, they are generative mechanisms accepted as valid as long as the criterion of validation in which they are embedded is fulfilled."

"What do we explain?
We explain our experiences. . . ."

"What do we explain?
We explain our experiences
with the coherences of our experiences.
We explain our living with the coherences of our living.

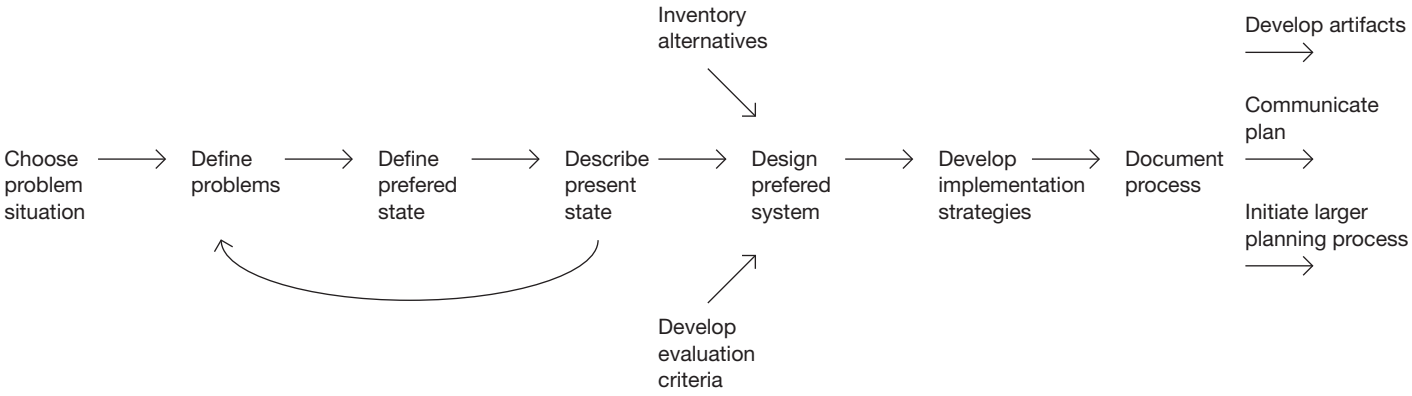
Explanations are not so in themselves;
explanations are interpersonal relations."

CVSE	**Scientific method**
1. The description of what an observer should do to live the experience to be explained.	1. The description of the phenomenon to be explained.
2. The proposition of a generative mechanism.	2. The proposition of the hypothesis or model of the reality that underlies the phenomenon being explained.
3. The deduction from all the experiential coherences of the observer implied in 2 of other possible experiences that he or she may live and of what he or she should do to have them.	3. The prediction of other phenomena assuming that 2 is valid together with the proposition of what to do to show them.
4. Doing what has been deducted in 3, and if the observer lives the experiences there deduced, then 2 is accepted as scientific explanation.	4. Doing what has been deduced in 3, and if the predicted new phenomena occur, the hypothesis or model presented in 2 is confirmed.

Comprehensive anticipatory design science
after Buckminster Fuller (1950?)

According to the Buckminster Fuller Institute, Fuller began formulating his theory of a comprehensive anticipatory design science as early as 1927. In 1950, he outlined a course, which he taught at MIT in 1956 as part of the Creative Engineering Laboratory. Students included engineers, industrial designers, materials scientists, and chemists, representing research and development corporations from across the country.

The assertion that design is a science was most powerfully articulated by Carnegie vv Herbert Simon (1969) in *The Sciences of the Artificial*. Simon's view is no longer fashionable. Most academic designers remain within Schools of Art. Some, such as Banathy (1996), suggest design is a third way of knowing distinct from the humanities and the sciences.

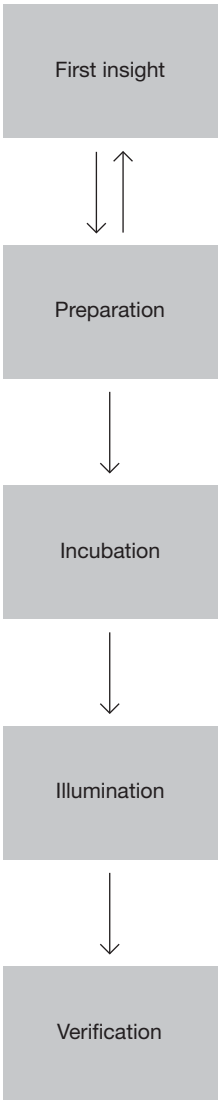


Design process and practice
after Richard Buchannan (1997)

Buchannan has a PhD in rhetoric and has taught design for many years—also at Carnegie Mellon. Below, he provides a practical model for students. Note the repetition of research, scenario building, and visualization in the three middle phases.

	Phases	Objectives	Characteristic activities	
0.	Vision & strategy	Discover governing ideas and circumstances - identify organizational vision & strategy - prepare design brief	Dialogue Strategic planning Strategic design planning, with vision Of the product development process	
1.	Brief	Indentification and selection - identify and select the initial issues, function, and features to be addressed	Discussion Research Scenario building Visualization Project planning Documentation	Observation, etc. Scribing Concept mapping
2.	Conception	Invention and judgment - invent possible concepts of the product - judge which concepts are viable	Research Brainstorming Scenario building Early & frequent visualization Documentation	Observation Concept mapping Sketching Modelling
3.	Realization	Disposition and evaluation - plan and make prototype of the product - evaluate by user testing	Research Scenario building & refinement Visualization Construction Documentation	 Prototype Evaluate Prototype Evaluate Prototype Evaluate
4.	Delivery	Presentation - present prototype, documentation, and production specifications	Oral presentation Written presentation Prototype demonstration	

Creative process
after Bryan Lawson (1980)



Lawson, an architect, compares the creative process to the design process. “The period of ‘first insight’ (Kneller 1965) simply involves the recognition that a problem exists and a commitment is made to solving it. This period may itself last for many hours, days or even years. The formulation of the problem may often be a critical phase in design situations. As we have seen, design problems are rarely initially entirely clear and much effort has to be expended in understanding them thoroughly.

The next phase of ‘preparation’ involves much conscious effort to develop an idea for solving the problem. (MacKinnon 1976) As with our maps of the design process it is recognized that there may be much coming and going between these first two phases as the problem is reformulated or even completely redefined.

Yet all these writers emphasize here that this period of preparation involves deliberate hard work and is then frequently followed by a period of ‘incubation’ which involves no apparent effort, but which is often terminated by the emergence of an idea (‘illumination’).

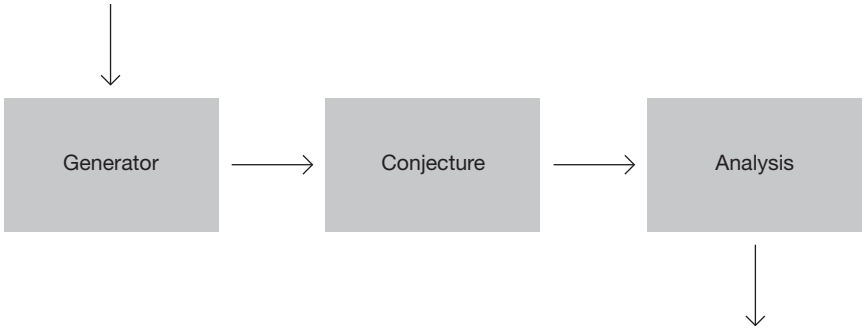
Some authors (MacKinnon 1976) explain this as unconscious cerebration during the incubation period. The thinker is unwittingly reorganizing and re-examining all his previous deliberate thoughts. Other writers suggest that by withdrawing from the problem the thinker is then able to return with fresh attitudes and approaches which may prove more productive than continuing his initial thought development.

Once the idea has emerged all writers agree upon a final period of conscious verification in which the outline idea is tested and developed.”

Primary generator
after Jane Darke (1978)

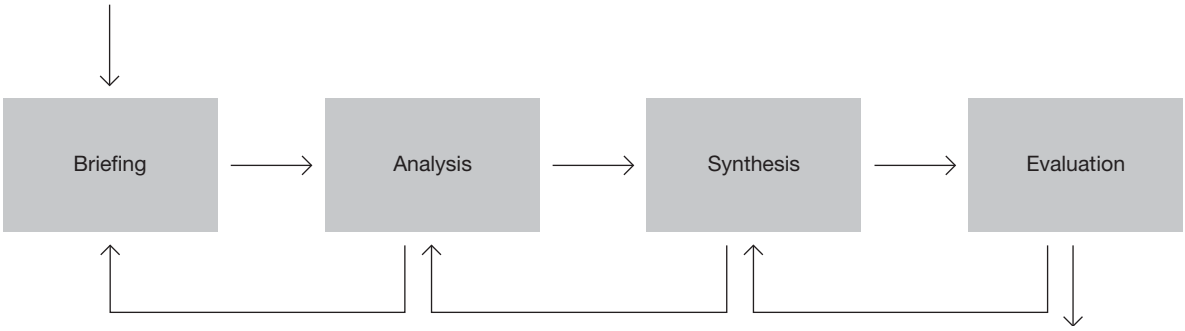
Lawson (1990) reports on Darke's finding that at least some architects begin the design process with a simple idea or "primary generator". "Thus, a very simple idea is used to narrow down the range of possible solutions, and the designer is then able rapidly to construct and analyze a scheme. Here again, we see this very close, perhaps inseparable, relation between analysis and synthesis."

Lawson suggests Darke's model was anticipated by Hiller et al (1972). Lawson summarizes Darke's model, "In plain language, first decide what you think might be an important aspect of the problem, develop a crude design on this basis and examine it to see what else you can discover about the problem." Note the similarity to "hacking" in software development.



Design process
after Jane Darke (1978)

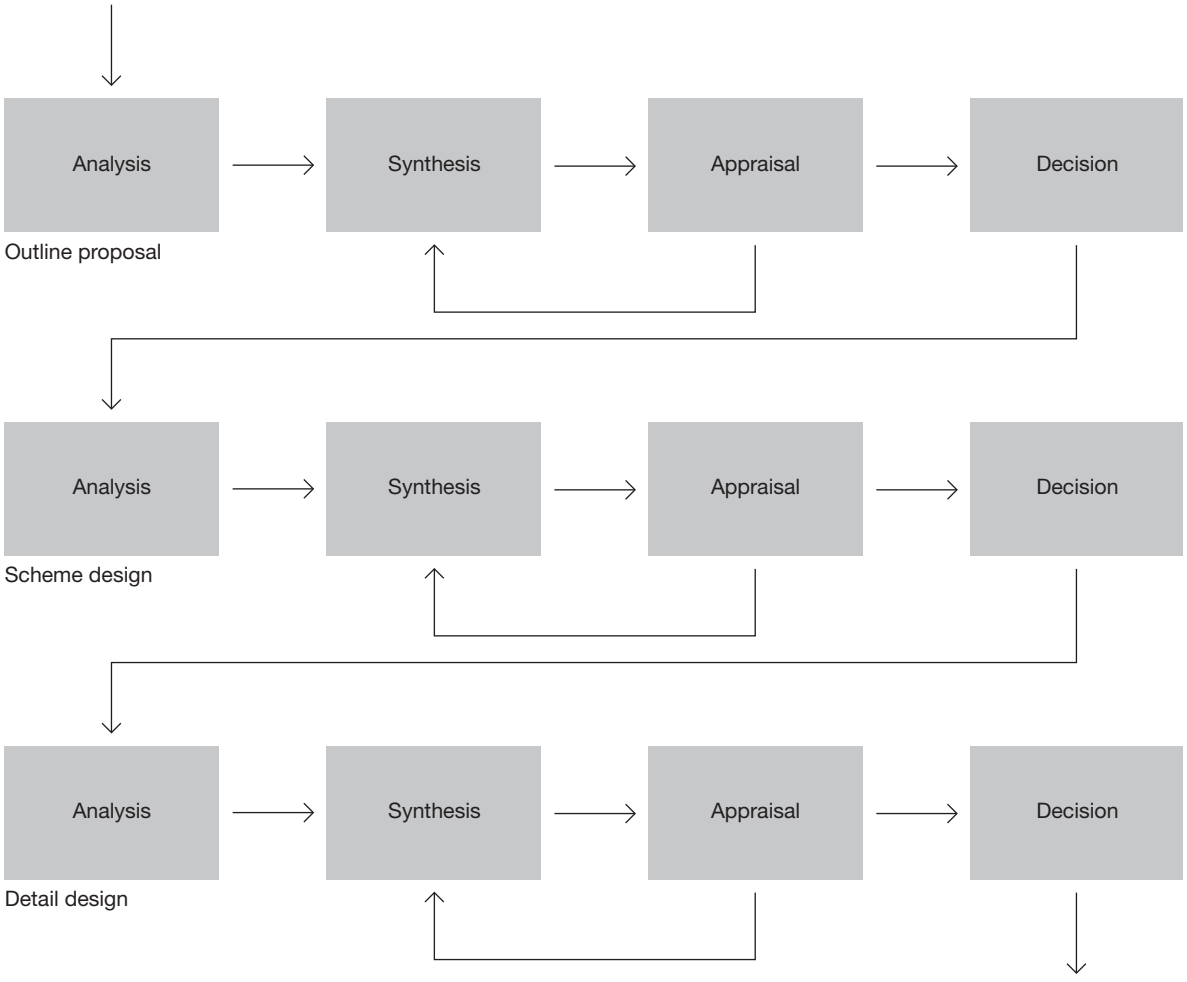
Based on Darke's research, Lawson suggests a looping relationship between brief and analysis. One of the architects Darke interviewed described the process, ". . . a brief comes about through essentially an ongoing relationship between what is possible in architecture and what you want to do, and everything you do modifies your idea of what is possible . . . you can't start with a brief and [then] design, you have to start designing and briefing simultaneously, because these two activities are completely interrelated." (For another take on this idea, see page 26.) Lawson points out that this may be one reason "clients often seem to find it easier to communicate their wishes by reacting to and criticizing a proposed design, than by trying to draw up an abstract comprehensive performance specification."



Design process

after Thomas A. Marcus (1969) and Thomas W Maver (1970)

Typically, in design process models evaluation follows analysis and synthesis. Marcus and Maver substitute decision, casting the design process as a series of decisions. They layer these decisions in three levels, outline proposals, scheme design, and detail design. This iterative structure is similar to that proposed by Banathy (1996) and Cross (2000). (See pages 24 and 25.) It's also similar to Boehm's spiral. (See page 122.) The three-level, four-step structure of this model anticipates the structure of the AIGA model on the next spread.



AIGA

Process of designing solutions
after Clement Mok and Keith Yamashita (2003)

American Institute of Graphic Arts (AIGA) president Clement Mok enlisted Keith Yamashita to help the organization help graphic designers explain what they do. Mok and Yamashita produced a cheery little book describing a 12-step process in which designers are “catalysts” for change.

The book casts design in terms of problem solving, yet it also promises innovation.

Defining the problem

1	2	3	4
Defining the problem	Envisioning the desired end state (knowing what victory looks like)	Defining the approach by which victory can be achieved	Inciting support and then action

Innovating

5	6	7	8
Seeking insight to inform the prototyping of the solution	Prototyping potential solutions	Delineating the tough choices	Enabling the team to work as a team

Generating value

9	10	11	12
Choosing the best solution then activating it	Making sure people know about your solution	Selling the solution	Rapidly learning and “tacking” based on your successes and failures

Case study, using the AIGA process in Iraq
by Nathan Felde

AIGA has tried to use its 12-step model as a structure for organizing case studies. Nathan Felde provided an example.

Defining the problem

1	2	3	4
Defining the problem	Envisioning the desired end state (knowing what victory looks like)	Defining the approach by which victory can be achieved	Inciting support and then action
Need oil.	**Got oil.**	**Control Iraq.**	**Create “axis of evil” and climate of fear.**

Innovating

5	6	7	8
Seeking insight to inform the prototyping of the solution	Prototyping potential solutions	Delineating the tough choices	Enabling the team to work as a team
Ask dad.	**Grenada, Panama, Kuwait, Afghanistan . . .**	**Inspections versus overwhelming force.**	**“You are either with us or against us.”**

Generating value

9	10	11	12
Choosing the best solution then activating it	Making sure people know about your solution	Selling the solution	Rapidly learning and “tacking” based on your successes and failures
War. “Bombs away.”	**Call Rupert. (and Halliburton)**	**Weapons of mass destruction?**	**Liberating the people of Iraq**

What the AIGA didn't tell you
by Nathan Felde

Felde also offered an alternative version of the 12-step process, acknowledging aspects of the AIGA's function (and that of other professional organizations) which few bring up in public.

Discovering the opportunity

1	2	3	4
Acquiring a distinctive persona	Practicing charm until charisma is achieved	Circulating amongst the right people	Listening for developing changes in status quo

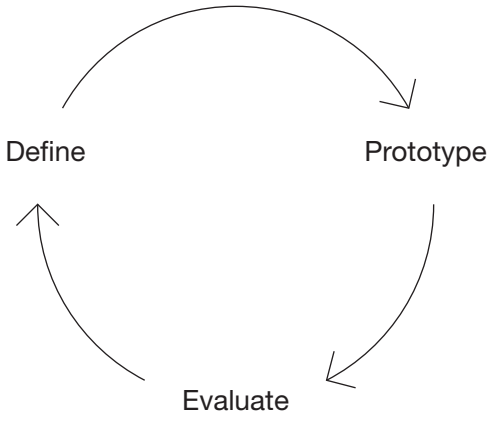
Obtaining the contract

5	6	7	8
Raising questions that only you can answer	Eliminating options other than your services	Presenting the proposal while appearing unavailable	Promising more than the contract requires

Getting paid

9	10	11	12
Discovering insurmountable difficulties	Finding someone on client side to blame	Rescuing client with bigger proposal	Announcing success with first stage

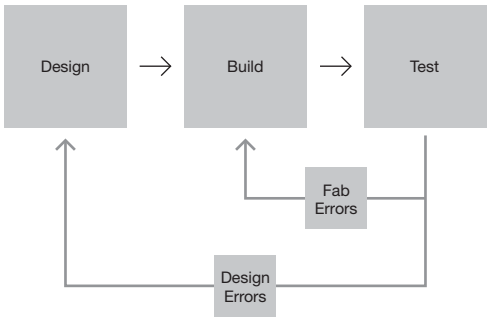
Alice Agogino



Design, build, test
after Alice Agogino (1 of 3)

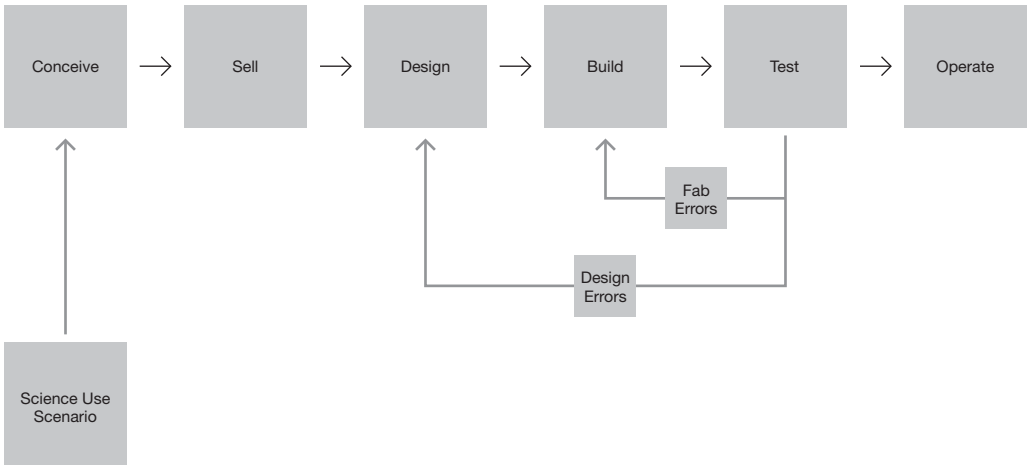
This model is the first in a series of three developed by Alice Agogino for NASA's Jet Propulsion Laboratory (JPL) at California Institute of Technology. Agogino is a professor of mechanical engineering at UC Berkeley.

In the first step, Agogino presents a variation on the classic goal-action feedback loop. (See page 117.)
Of course, design-build-test is also analogous to define-prototype-evaluate. (See facing page.)



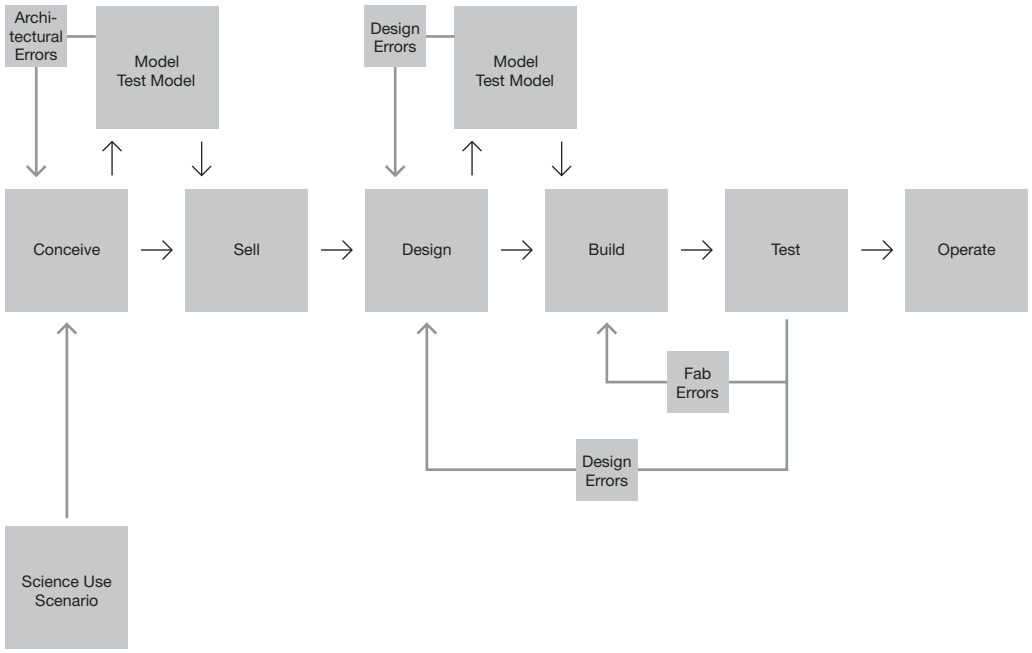
Design, build, test
after Alice Agogino (2 of 3)

In the second step, Agogino places the original design-build-test process in the context of a larger project.



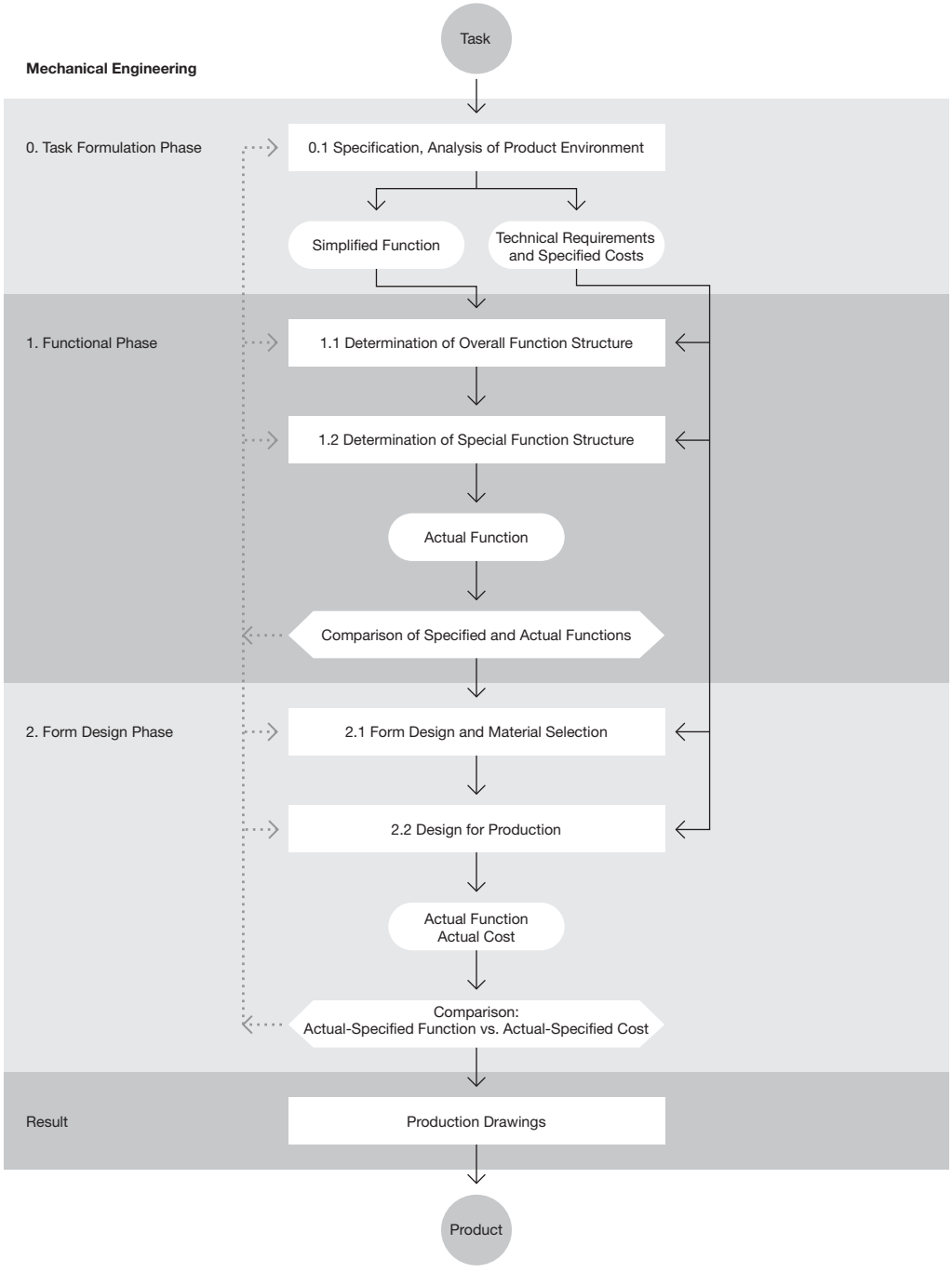
Design, build, test
after Alice Agogino (3 of 3)

In the last step, Agogio adds feedback loops with early tests of models in order to “find errors faster.”



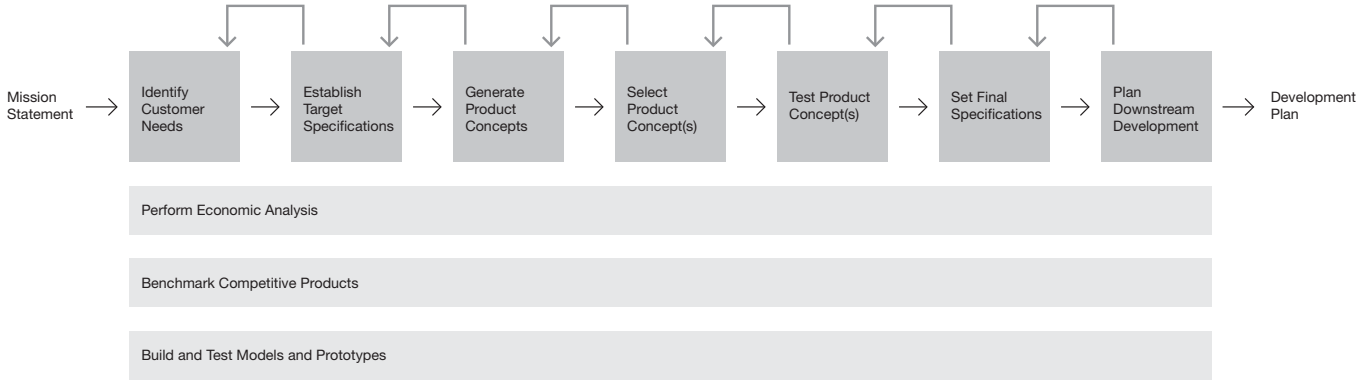
Mechanical engineering design process
after students at UC Berkeley Institute of Design (BID)

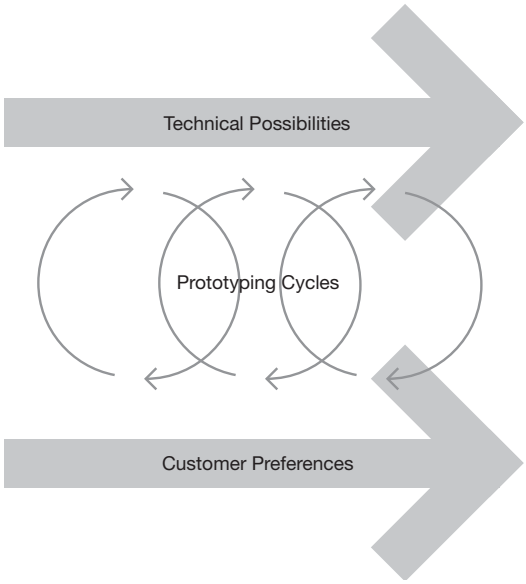
Agogino sometimes asks her students to diagram the design process—an interesting way to begin to understand how students (and others) understand things. Below is an example from one of her classes.



New product development process
after Steven D. Eppinger and Karl T. Ulrich (1995)

Alice Agogino introduced me to Eppinger and Ulrich's model of the product development process. It provides a useful outline, but does not capture the "messy" iteration typical of much product development work.



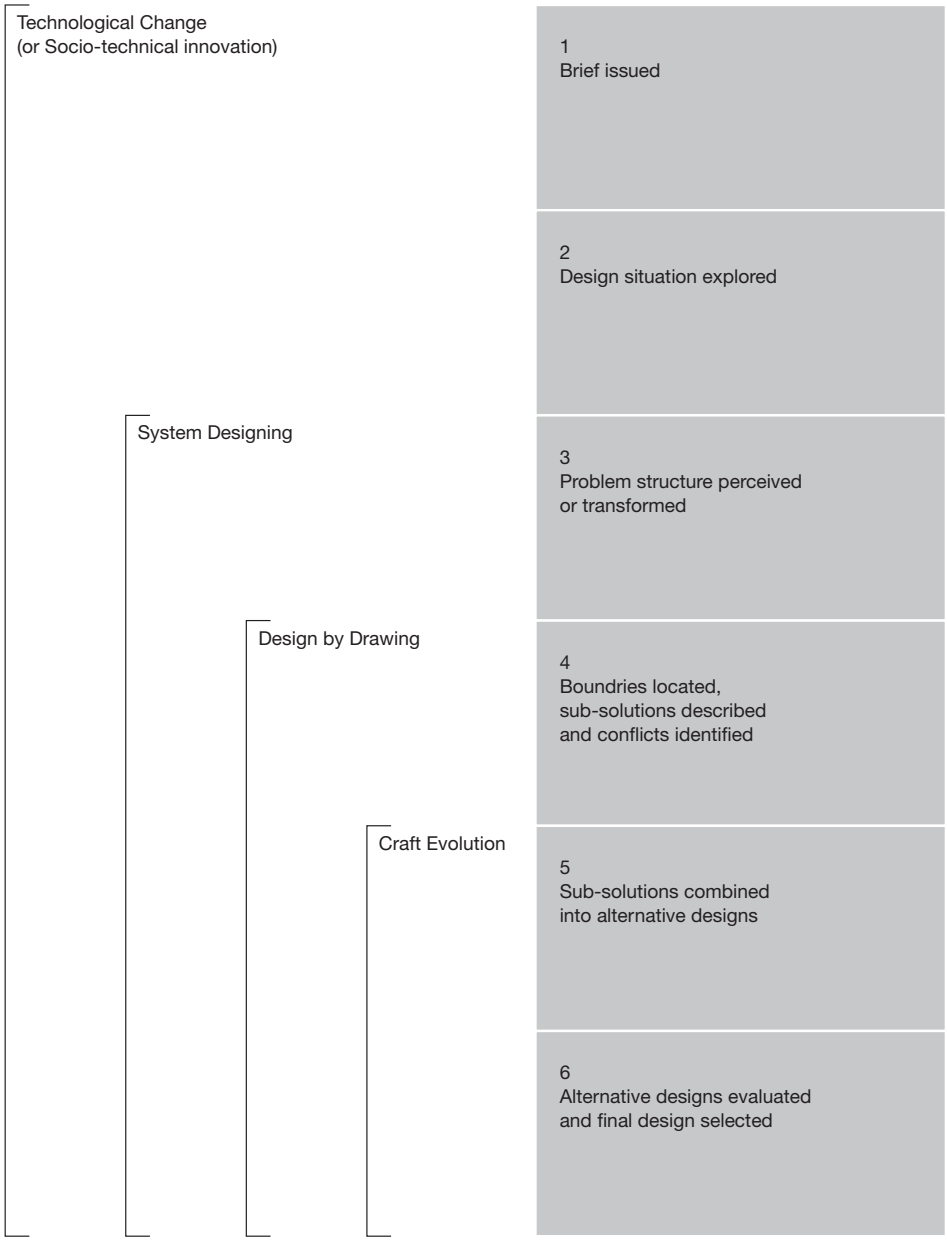


John Chris Jones

Design process
after John Chris Jones (1970)

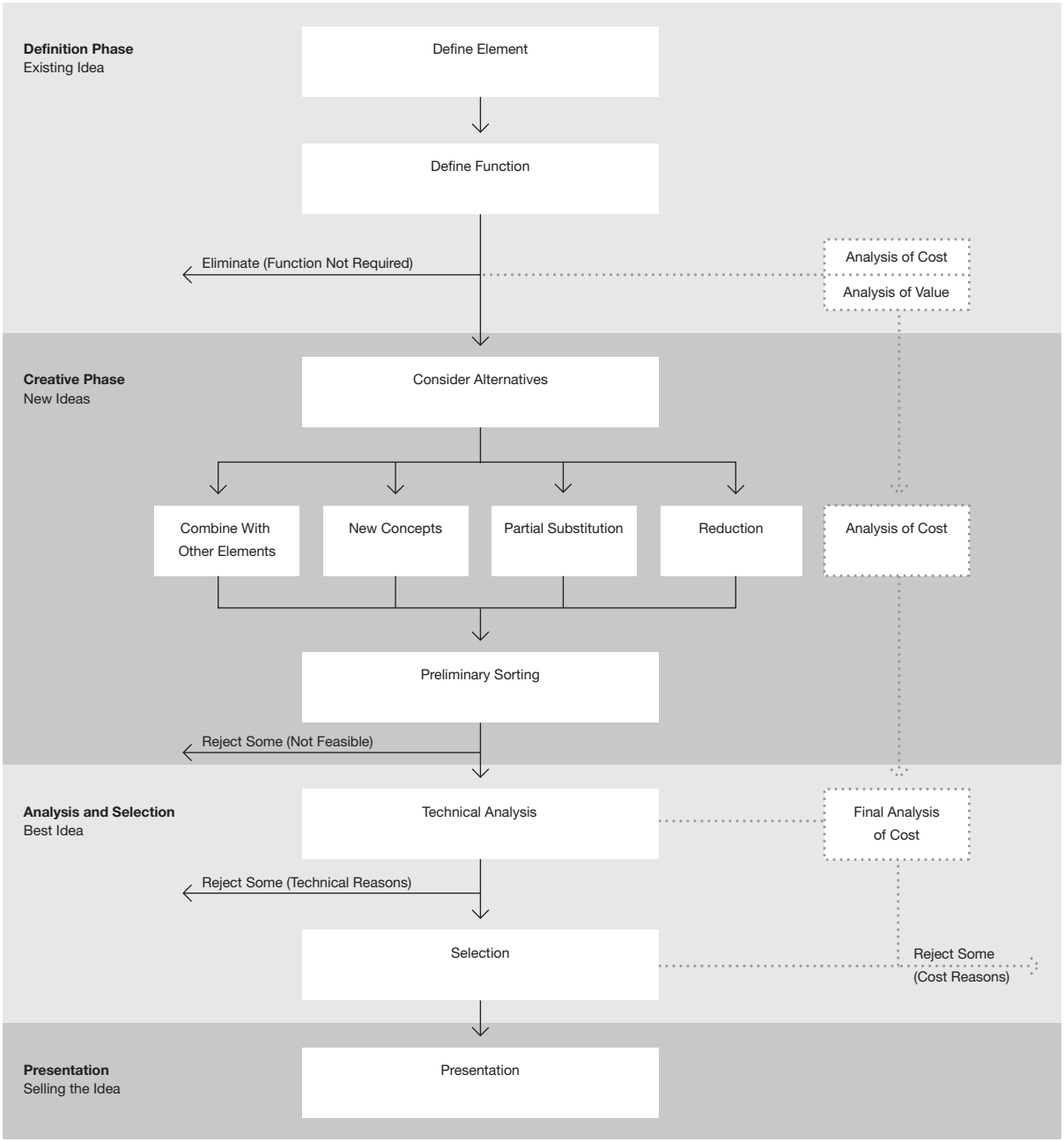
Along with Christopher Alexander and Bruce Archer, John Chris Jones was one of the pioneers of the design methods movement. Jones first published Design Methods in 1970. He included several models of design and the design process. I have included three in this section.

Jones used the model below for classifying and selecting design methods. Designers might use one or more methods to move from one step to another. Jones notes that the steps decrease in generality and increase in certainty. Jones also provides a scale for describing the applicable range of a method. (See the left side of the diagram.) We may also apply his scale to the scope of problem being undertaken. In this way, Jones's scale is similar to the models of design scope described by Doblin and Alexander. (See pages 17-18.)



Value analysis
after John Chris Jones (1970)

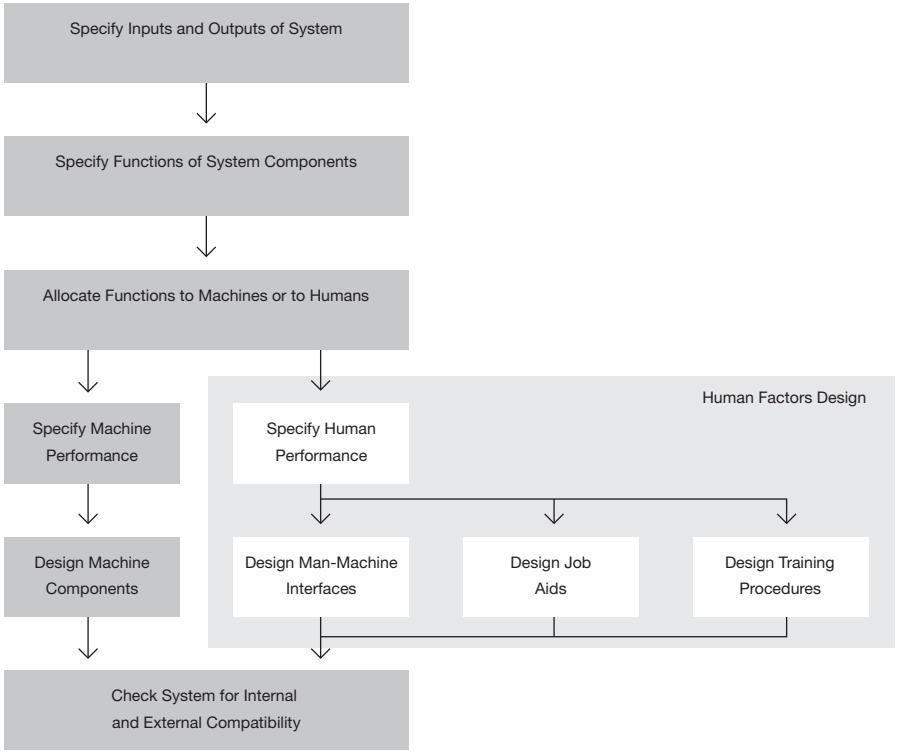
Jones described value analysis as a design method, one aimed "to increase the rate at which designing and manufacturing organizations learn to reduce the cost of a product." He saw it as applying to the design of an element within a larger system. Yet his value analysis process (as he diagrammed it) is itself a sort of design process—albeit with a special emphasis on cost. This example of a design process-nested within a design process nicely illustrates the recursive nature of designing.



Man-machine system designing
after John Chris Jones (1970)

Jones also described man-machine system designing as a design method, one aimed “to achieve internal compatibility between the human and machine components of a system, and external compatibility between the system and the environment in which it operates.”

This method, too, is a sort of design process. Jones notes “the diagram should not be taken to imply a linear sequence of stages. The specifications in each box can be attended to in any order and will require many cross-references before they are complete.” He suggests deliberately reversing “the traditional sequence of machine-first-people-second” design. He proposes beginning with training procedures, working out the man-machine interface, and then designing the machine to support the desired training and interface.



Eight phases of a project

Sometimes presented as six phases of a project

People have passed variations of this project parody around for years. Lawson sites an example seen on a wall of the Greater London Council Architects Department in 1978. More recently, Harold Kerzner offered the variation below. One reason these parodies are popular may be that they contain a large measure of truth.

1. Project Initiation
2. Wild Enthusiasm
3. Disillusionment
4. Chaos
5. Search for the Guilty
6. Punishment of the Innocent
7. Promotion of Non-Participants
8. Definition of Requirements

Consultant models

A few consultancies publish their processes.

Some firms see their processes as a competitive advantage and thus keep them proprietary.

Some firms operate without processes, but who would admit such a thing?

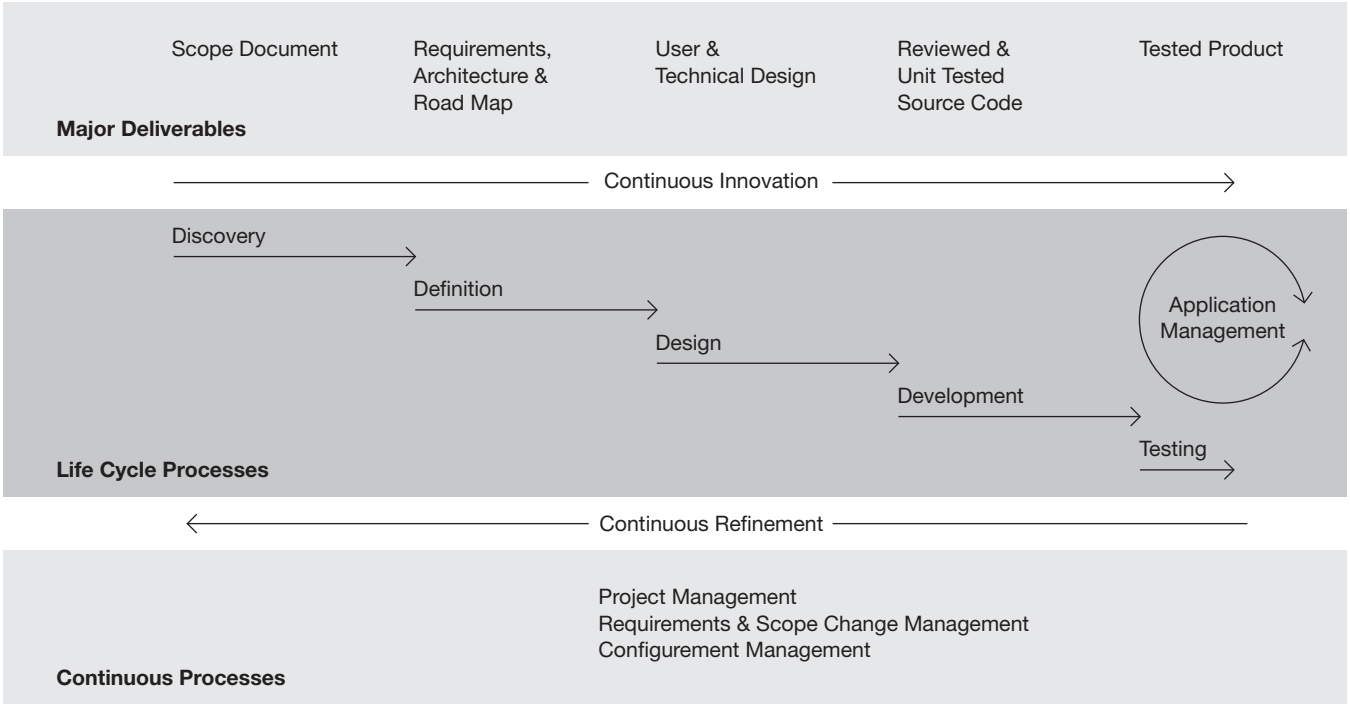
4D software process
and variations

The 4D software process, (define, design, develop, deploy) gained wide popularity among consultants developing web-sites during the internet boom. One company, Information Systems of Florida (SF) claims to have trademarked the four steps. The phrase is useful as a mnemonic device, but the wide range of variations suggests something is missing, for example, feedback and iteration. Author and date unknown.

		Define	Design	Develop	Deploy			
		Define	Design	Develop	Deploy	Debrief		Imirage
		Define	Design	Develop	Deploy	Dedicate		Bonns
		Define	Design	Develop	Deploy	Do Business		Q4-2
		Define	Design	Develop	Deploy	Enhance		Satoria
		Define	Design	Develop	Deploy	Maintain		Chris Brauer
	Diagnose	Define	Design	Develop	Deploy			Muirmedia
	Discover	Define	Design	Develop	Deploy			D5tech et al.
	Discover	Define	Design	Develop	Deploy	Defend		Dillon Group
	Discover	Define	Design	Develop	Deploy	Denouement		Cris Ippolite
Engagement	Discovery	Define	Design	Develop	Deploy			Team 1
	Plan	Define	Design	Develop	Deploy	Conclude		Proxicom
	Analyze	Define	Design	Develop	Deploy	Assess	Maintain	Hbirbals
		Define	Design	Develop	Test	Deploy	Manage	Borland
Inform	Define	Detail	Design	Develop	Deploy			Phoenix Pop

IT consulting process overview
after Mindtree Consulting

Mindtree places the 4D process in a larger context, linking each step to deliverables and related processes. The pairing of process steps and deliverables in a matrix is an important and recurring framework or archetype.



Other models

This page presents a sampling of design process models from leading consultancies. They resemble the 4D model.

On the facing page is IDEO's design process as described in Business Week. IDEO is a large (by design standards) multidisciplinary design consultancy.

Studio Archetype, 1998

Definition	Concept	Creation	Implementation

Cheskin, 2004

Discover	Identify	Validate	Articulate

Frog, 2004

Product Lifecycle Phases

Conceptual Design	Detailed Design	Procurement/Production	Operations/Support	

Product Design

Project Definition	Product Definition	Product Development	Product Engineering	Production

Brand & Space Process

Investigation	Concept Development	Concept Refinement/Validation	Implementation

Digital Media Process

Investigation	Exploration	Definition	Implementation	Integration/Testing	Launch

IDEO (2004)

1. Observation
IDEO's cognitive psychologists, anthropologists, and sociologists team up with corporate clients to understand the consumer experience.

Some of IDEO's techniques:

Shadowing Observing people using products, shopping, going to hospitals, taking a train, using their cell phones.
Behavioral mapping Photographing people within a space, such as a hospital waiting room, over two or three days.
Consumer journey Keeping track of all the interactions a consumer has within a product, service, or space.
Camera journals Asking consumers to keep visual diaries of their activities and impressions relating to a product.
Extreme user interviews Talking to people who really know—or know nothing—about a product or service, and evaluating their experience using it.
Storytelling Prompting people to tell personal stories about their consumer experiences.
Unfocus groups Interviewing a diverse group of people: To explore ideas about sandals, IDEO gathered an artist, a bodybuilder, a podiatrist, and a shoe fetishist.

2. Brainstorming
An intense idea-generating session analyzing data gathered by observing people. Each lasts no more than an hour. Rules of brainstorming are strict and are stenciled on the walls.

Defer judgment Don't dismiss any ideas
Build on the ideas of others No "buts," only "ands."
Encourage wild ideas Embrace the most out-of-the-box notions because they can be the key to solutions.
Go for quantity Aim for as many new ideas as possible. In a good session, up to 100 ideas are generated in 60 minutes.
Be visual Use yellow, red, and blue markers to write on big 30-inch by 25-inch Post-its that are put on a wall.
Stay focused on the topic Always keep the discussion on target.
One conversation at a time No interrupting, no dismissing, no disrespect, no rudeness.

3. Rapid prototyping
Mocking-up working models helps everyone visualize possible solutions and speeds up decision-making and innovation.

Some guidelines:

Mock-up everything It is possible to create models not only of products but also of services such as health care and spaces such as museum lobbies.
Use videography Make short movies to depict the consumer experience.
Go fast Build mock-ups quickly and cheaply. Never waste time on complicated concepts.
No frills Make prototypes that demonstrate a design idea without sweating over the details.
Create scenarios Show how a variety of people use a service in different ways and how various designs can meet their individual needs.
Bodystorm Delineate different types of consumers and act out their roles.

4. Refining
At this stage, IDEO narrows down the choices to a few possibilities.

Here's how it's done:

Brainstorm in a rapid fashion to weed out ideas and focus on the remaining best options.
Focus prototyping on a few key ideas to arrive at an optimal solution to a problem.
Engage the client actively in the process of narrowing the choices.
Be disciplined and ruthless in making selections.
Focus on the outcome of the process—reaching the best possible solutions.
Get agreement from all stakeholders. The more top-level executives who sign off on the solution, the better the chances of success.

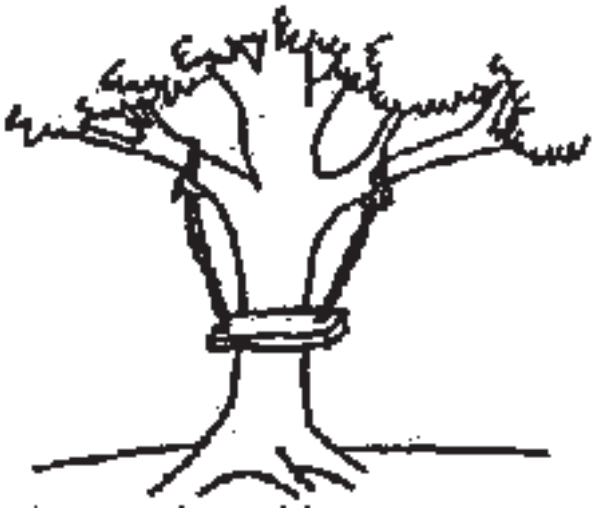
5. Implementation
Bring IDEO's strong engineering, design, and social-science capabilities to bear when actually creating a product or service.

Tap all resources Involve IDEO's diverse workforce from 40 countries to carry out the plans.
The workforce Employees have advanced degrees in different kinds of engineering: mechanical, electrical, biomechanical, software, aerospace, and manufacturing. Many are experts in materials science, computer-aided design, robotics, computer science, movie special effects, molding, industrial interaction, graphic and web information, fashion and automotive design, business, communications, linguistics, sociology, ergonomics, cognitive psychology, biomechanics, art therapy, ethnology, management consulting, statistics, medicine, and zoology.

As proposed by the project sponsor

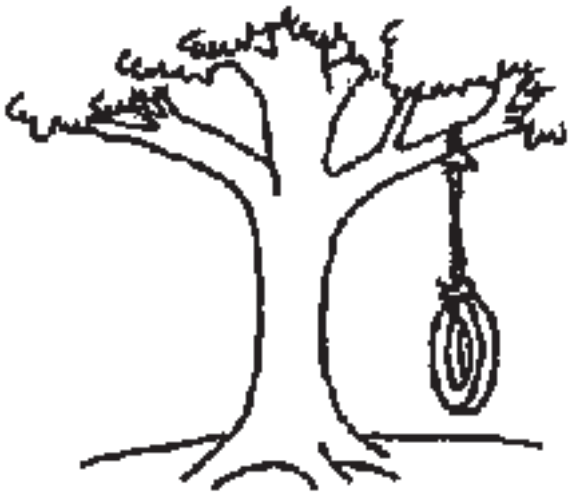
As specified in the project request

As designed by the senior analyst



As produced by the programmers

As installed at the user's site



What the user wanted

Software development models

Development processes remain a topic of heated discussion in the software development world.

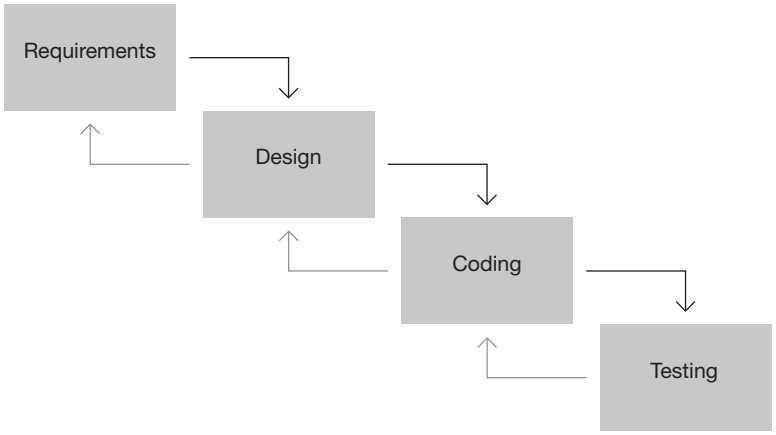
This section provides an overview of some of the prominent models.

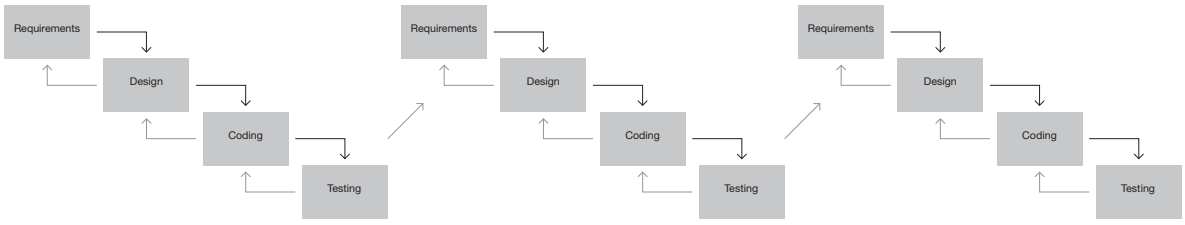
Waterfall lifecycle
after Philippe Kruchten (2004)

The essence of the “waterfall” approach is getting one stage “right” before moving on to the next. Output (a “deliverable document”) from one phase serves as input (requirements) to the next phase. Kruchten noted, “Of paramount importance for certain projects is the issue of freezing the requirements specifications (together with some high-level design) in a contractual arrangement very early in the lifecycle, prior to engaging in more thorough design and implementation work. This is the case when an organization has to bid a firm, fixed price for a project.” Per Kroll (2004) noted, “Many design teams would view modifying the design after Stage 1 as a failure of their initial design or requirements process.”

Kroll admitted, “In practice, most teams use a modified waterfall approach, breaking a project down into two or more parts, sometimes called phases or stages. This helps to simplify integration, get testers testing earlier, and provide an earlier reading on project status. This approach also breaks up the code into manageable pieces and minimizes the integration code . . .”

According to Kruchten, “we inherited the waterfall lifecycle from other engineering disciplines, where it has proven very effective. It was first formally described by Winston Royce in 1970.”



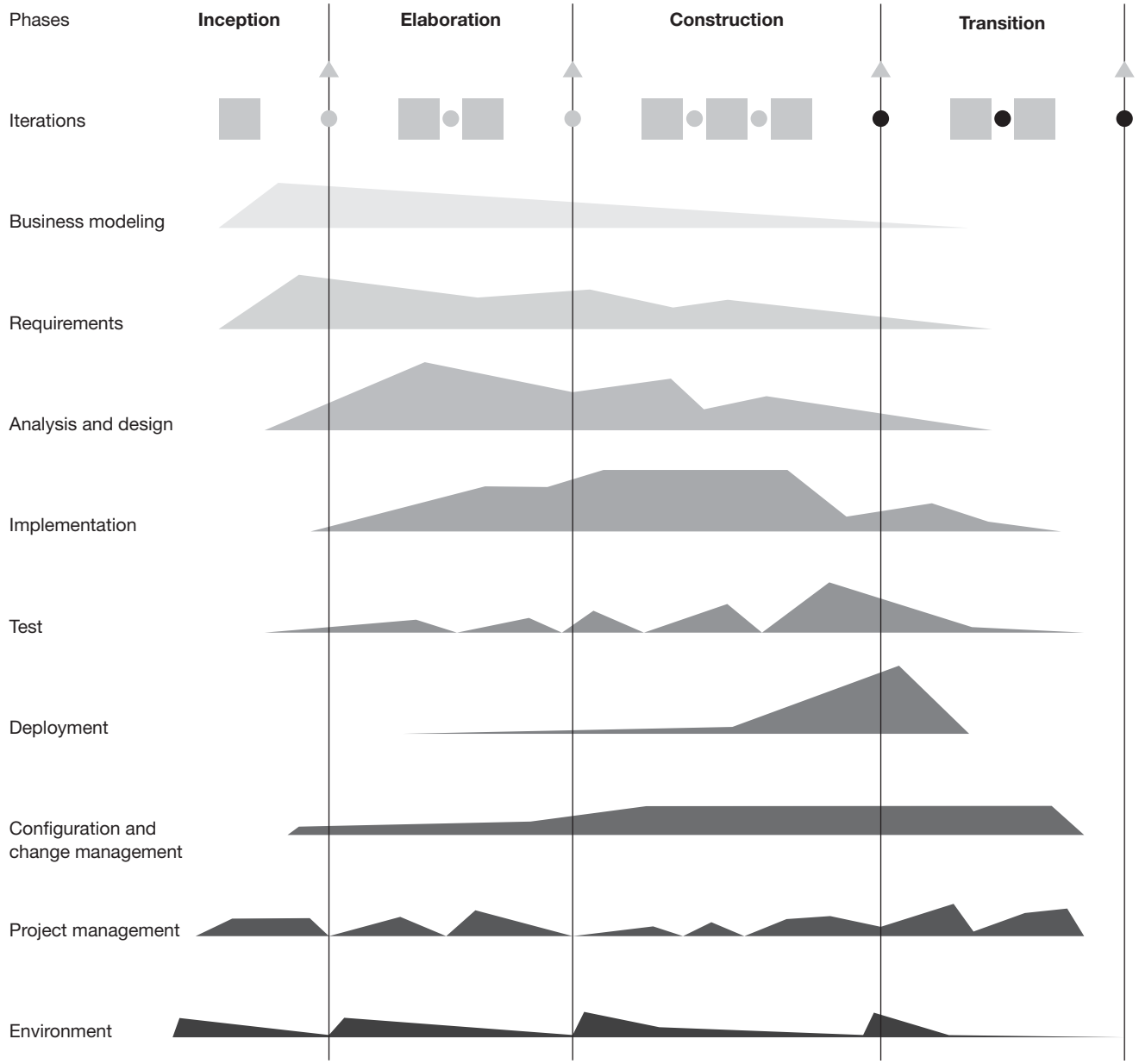


Rational Unified Process (RUP)
after Phillippe Kruchten (2003)

RUP follows an "iterative" lifecycle—as opposed to the "waterfall" lifecycle—"developing in iterations that encompass the activities of requirements analysis, design, implementation, integration, and tests. One of the best descriptions is in Professor Barry Boehm's paper on the "spiral" model. You can summarize it with the catch phrase, 'Analyze a little, design a little, test a little, and loop back.'" (For more on Boehm's model, see page 122.)

Kruchten noted, "The process has two structures or, if you prefer, two dimensions:
- The horizontal dimension represents time and shows the lifecycle aspects of the process as it unfolds.
- A vertical dimension represents core process disciplines (or workflows), which logically group software engineering activities by their nature."

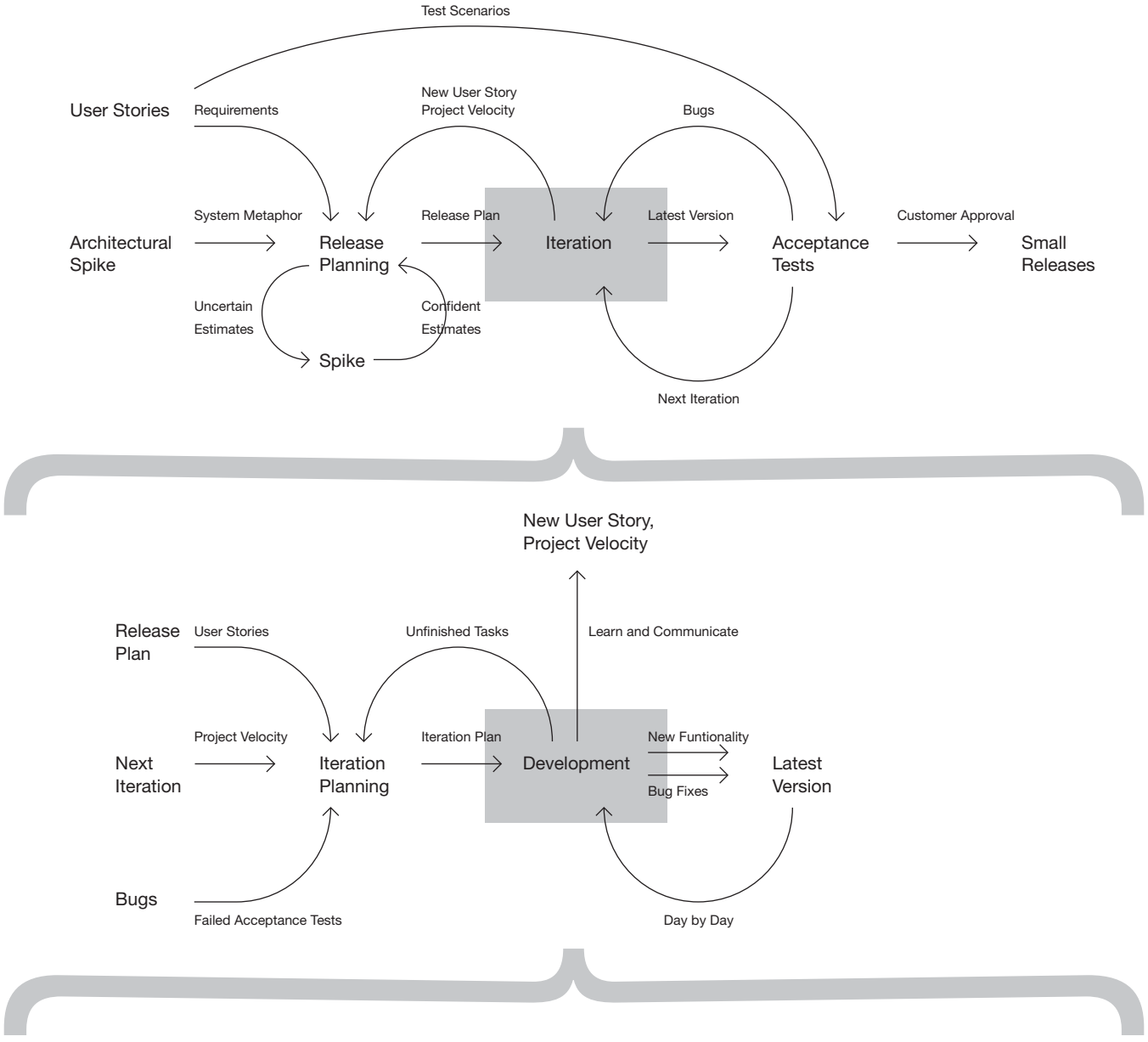
Rational Software was an independent developer purchased by IBM in 2003. Rational (and later IBM) developed and sold a suite of software development tools built around the Rational Unified Process (RUP). RUP was designed using the Unified Modeling Language (UML) and has as its underlying object model, the Unified Software Process Model (USPM).



Extreme Programming (XP) Process
after Don Wells (2000)

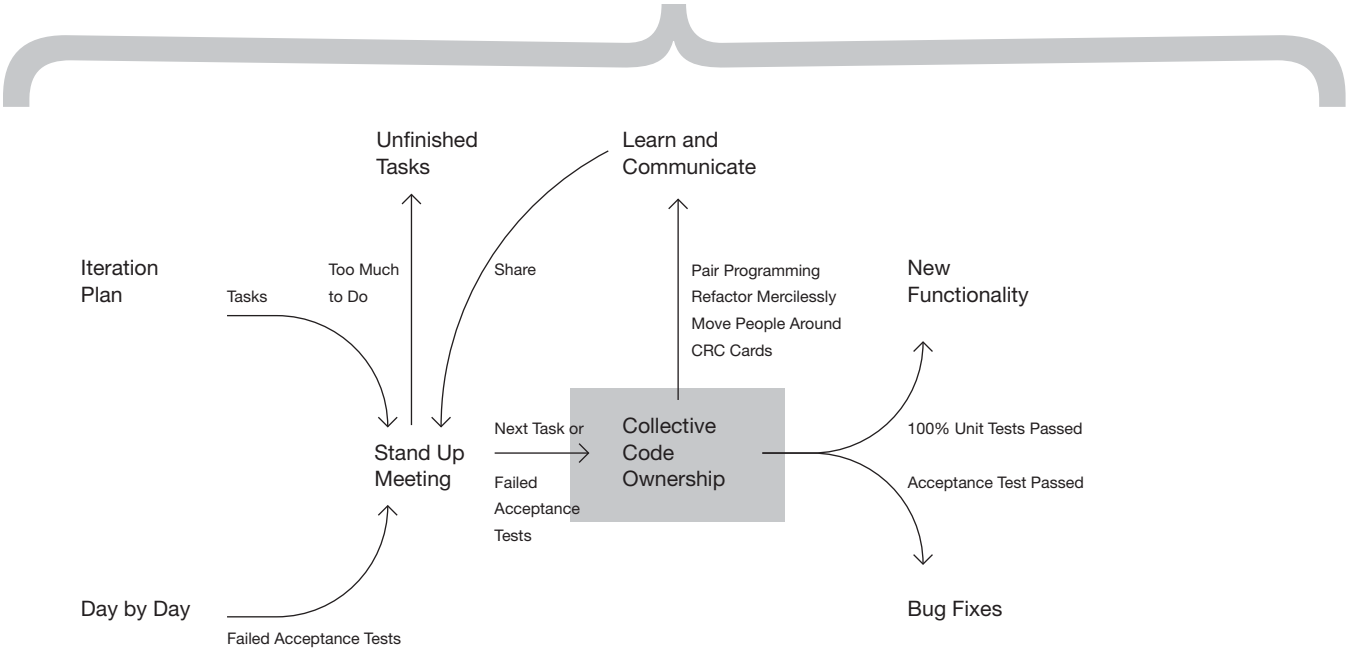
Kent Beck, founder of Extreme Programming, has described how he created XP in 1996. Chrysler asked him to put a payroll system project back on track. When they called him, eighteen months into the project, the system still couldn't print a check. Three weeks later, Beck had them print their first one. "Up until then I believed better programming would solve all the world's ills. Yes, you can screw up the programming so badly you kill the project. Usually, however, the problem concerns relationships between the business people and the programmers, the budget process, poor communications—factors unrelated to the programming. The context in which the software development takes place proves as important to the project's success as the programming itself."

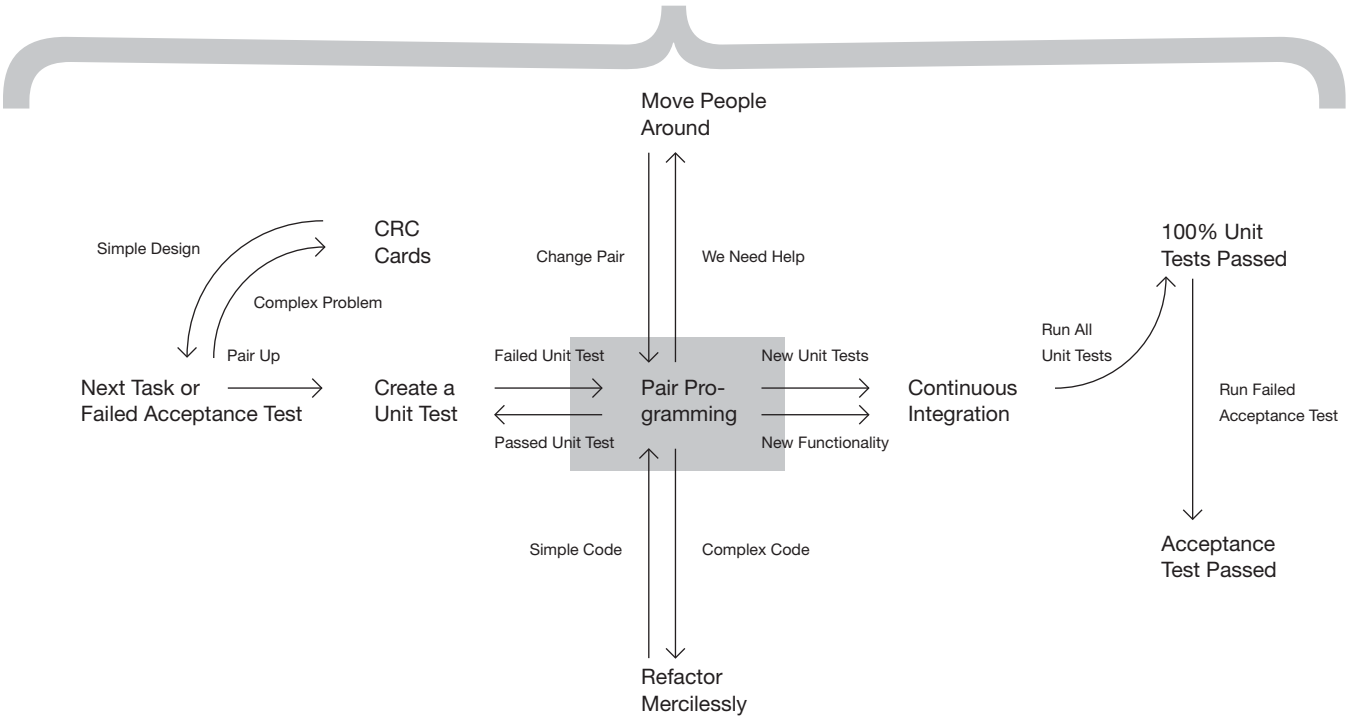
At its core, XP is a simple process of experimentation and improvement: Divide a project into "iterations"; in each iteration, implement a few new features called "stories"; for each story, write "acceptance tests" to demonstrate the story meets customer expectations. Alan Cooper, however, argues



XP is not a design process—because it includes no mechanism for understanding user goals. (For more on Cooper, see pages 86-91.)

The models below are nested. The first one shows the whole project; the second "zooms in" on iteration; the third "zooms in" on development; and the fourth on collective code ownership. At the center of the last diagram is pair programming, one of the primary distinguishing features of XP. Two programmers work together at a single computer. Beck claims this increases quality. It has to be a lot more fun than coding alone. (For another model of extreme programming, see page 127.)



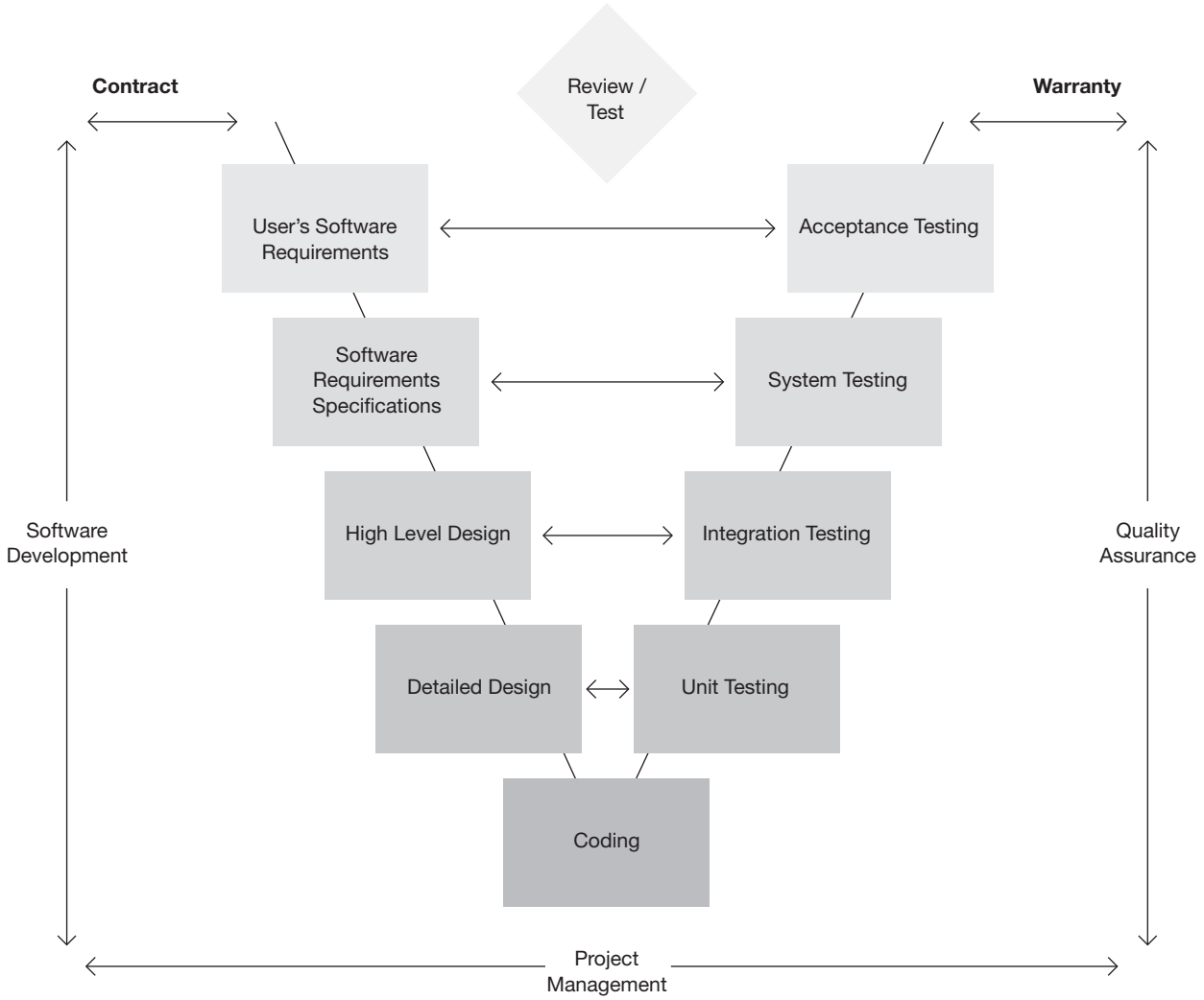


V model

Paul Rock (~1980), IABG (1992)

The principle characteristic of the V model seems to be that it weights testing equally with design and development. Goldsmith and Graham (2002) note, “In fact, the V Model emerged in reaction to some waterfall models that showed testing as a single phase following the traditional development phases . . . The V Model portrays several distinct testing levels and illustrates how each level addresses a different stage of the lifecycle. The V shows the typical sequence of development activities on the left-hand (downhill) side and the corresponding sequence of test execution activities on the right-hand (uphill) side”

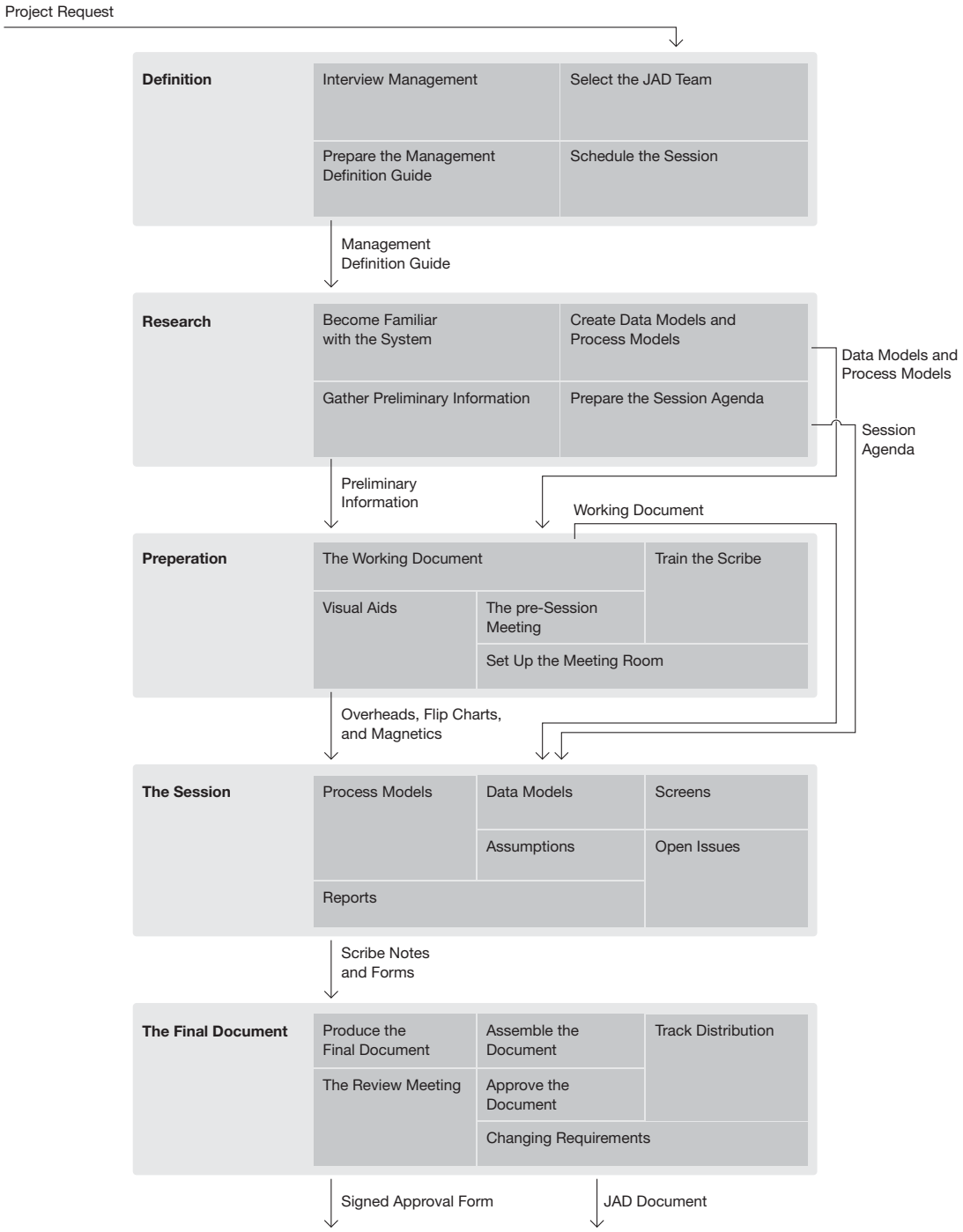
Accounts of this model’s origin vary. According to Goldsmith and Graham, “Initially defined by the late Paul Rook in the late 1980s, the V was included in the U.K.’s National Computing Centre publications in the 1990s with the aim of improving the efficiency and effectiveness of software development.” But according to Morton Hirschberg, formerly of the Army Research Laboratory, “The V Model is a series of General Directives (250, 251, and 252) that prescribe or describe the procedures, methods to be applied, and the functional requirements for tools to be used in developing software systems for the German Federal Armed Forces.”



Joint Application Development (JAD)
after Jane Wood and Denise Silver (1995)

JAD sessions (sometimes jam sessions) are intensive workshops, usually three to five days long, in which various levels of users meet with developers to hammer out requirements for a system. Typically consultants use the process to quickly lock down user requirements for automation projects—so they can minimize the time needed to define requirements and work within a fixed bid.

According to Carmel et al (1993), "JAD was conceived by Chuck Moris and Tony Crawford of IBM in 1977. The JAD approach was loosely derived from another IBM methodology—BSP (Business Systems Planning). The first JAD meetings . . . used the same basic JAD concepts still used today: user participant meetings, magnetic visual displays, and careful documentations of the meeting."



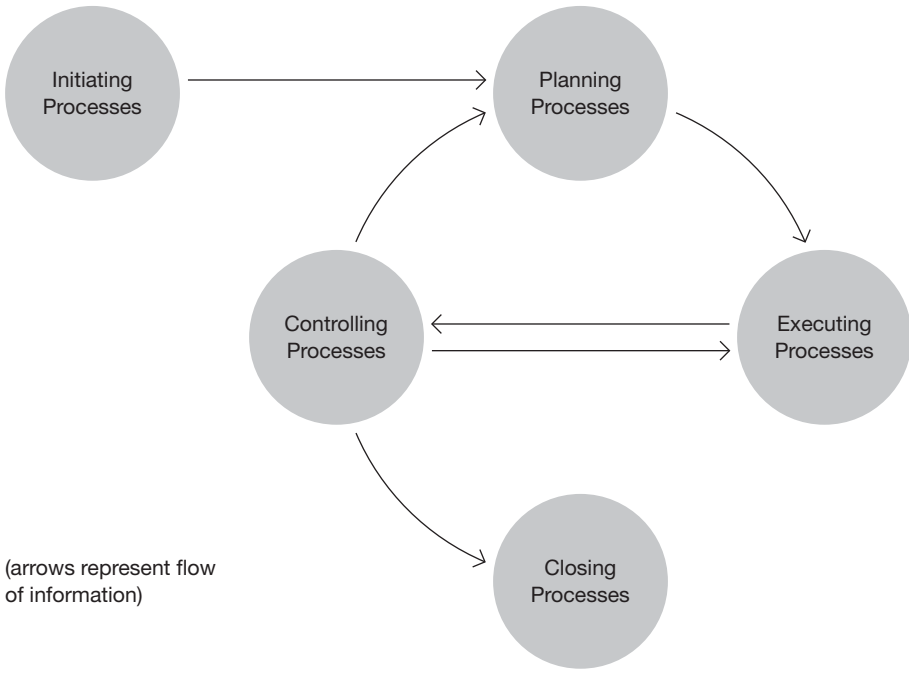
PMBOK (Project Management Body of Knowledge)

PMI (Project Management Institute) (1987)

Charbonneau wrote, “The PMBOK describes a set of generally accepted practices that PM practitioners can use to manage all types of projects, including software development and deployment projects.

The PMBOK defines a project as ‘a temporary endeavor undertaken to create a unique product or service.’ It defines PM as ‘the application of knowledge, skills, tools, and techniques to project activities to meet project requirements.’

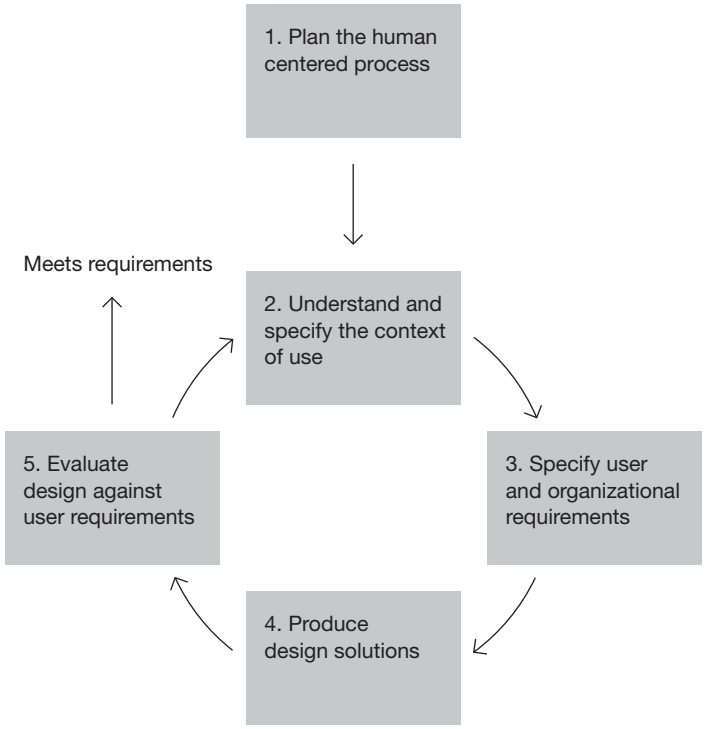
The PMBOK presents [thirty-nine] PM practices in logical groupings. One dimension describes [nine]‘knowledge areas’ while the other dimension describes project management processes split into five process groups.” The process groups are shown in the model below.



ISO 13407 Human-centered design processes for interactive systems

Tom Stewart et al. (1999)

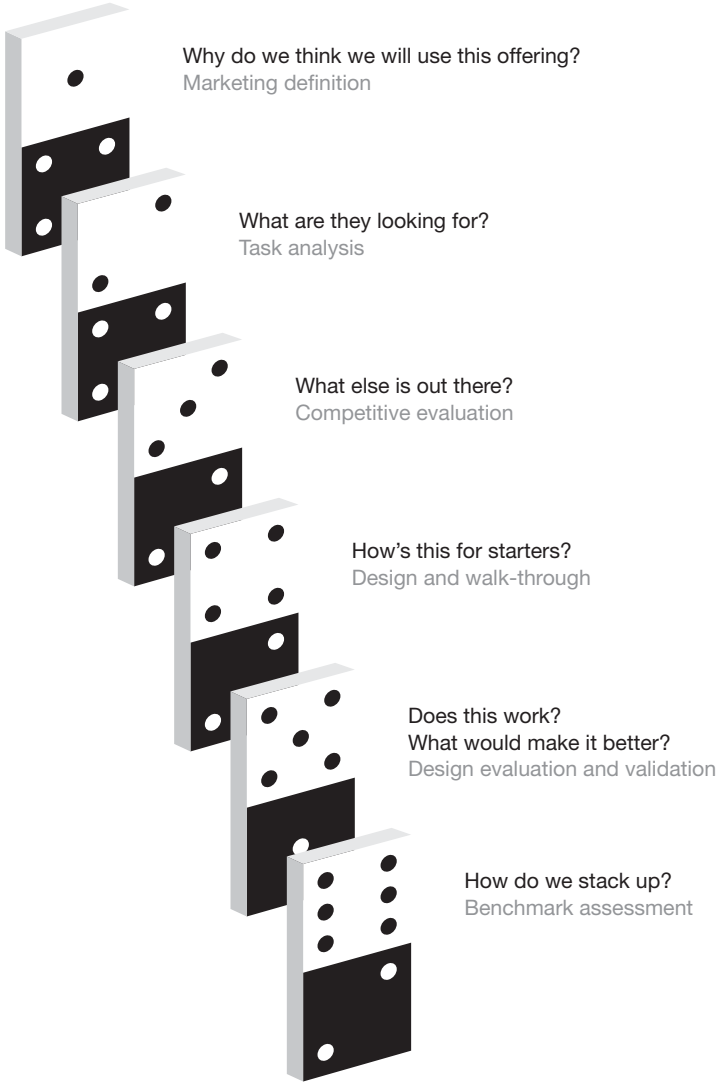
"ISO 13407 provides guidance on achieving quality in use by incorporating user centered design activities throughout the life cycle of interactive computer-based systems. It describes user centered design as a multi-disciplinary activity, which incorporates human factors and ergonomics knowledge and techniques with the objective of enhancing effectiveness and productivity, improving human working conditions, and counteracting the possible adverse effects of use on human health, safety and performance."



User-centered design process (UCD)
after Karel Vredenburg (2003)

Vredenburg describes this “simplified generic description of the design process” as follows: “The design process starts with the collection of relevant market definition information to answer the basic question, ‘Who do we think will use this offering?’ This involves understanding the target markets, types of users, prime competitors, market trends, high level needs and preferences, and so forth. Next, detailed information is collected from representative users within the target markets to understand their goals and tasks to answer the question, ‘What are they looking for?’ Following this, we attempt to understand how the tasks described in the prior step are carried out today either with a competitor’s product or an analog method. This answers the question, ‘What else is out there?’

At this point, conceptual design of the user experience starts, and early feedback is gathered from users, answering the question, ‘How’s this for starters?’ This leads to several cycles of iterative detailed design and user feedback through design evaluation and validation sessions, answering the questions, ‘Does this work?’ and ‘What would make it better?’ At the end of the development cycle, a user feedback benchmark assessment session is conducted to answer the question, ‘How do we stack up?’”



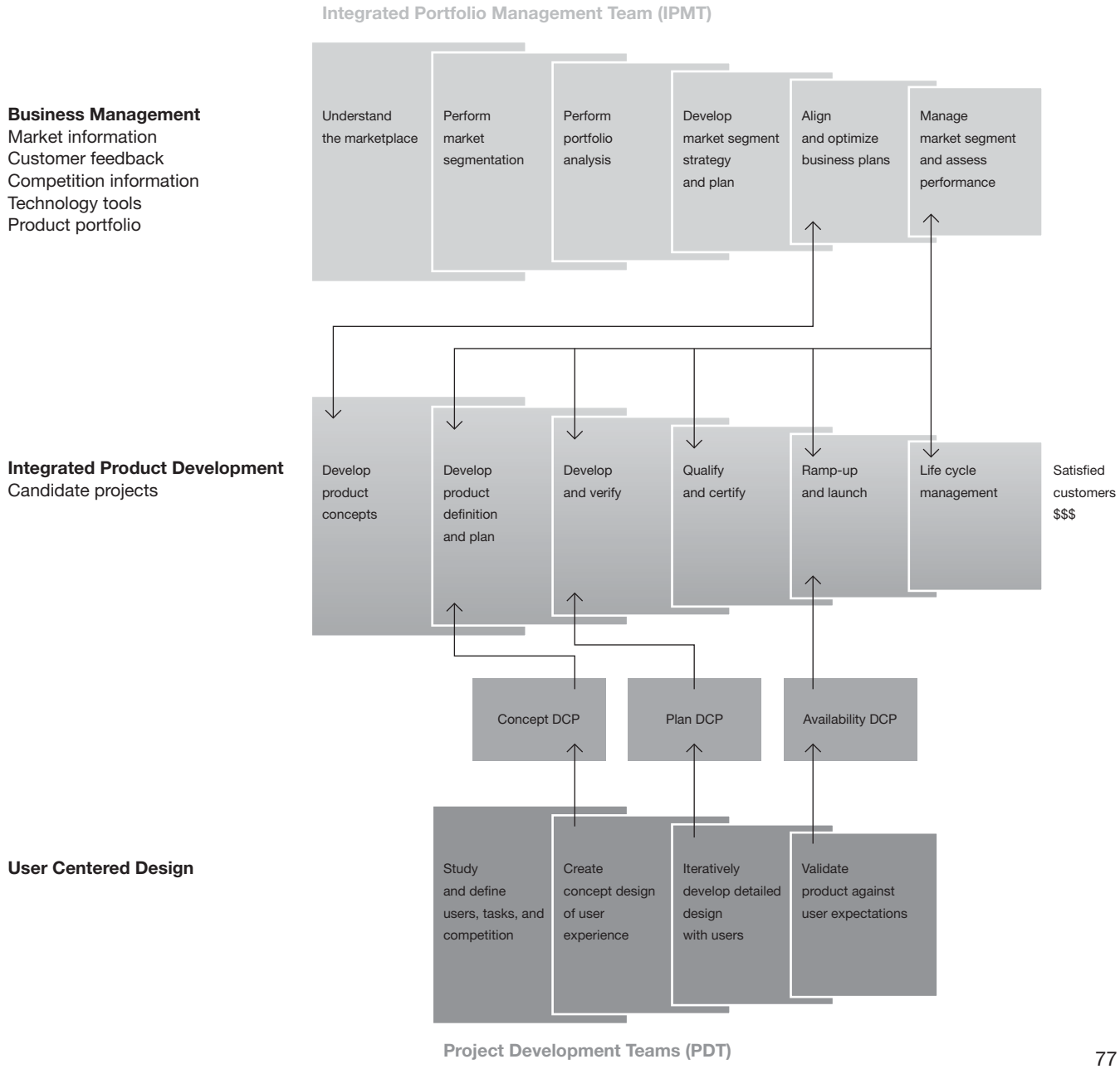
Relation of UCD to IPD and Business Management
after Karel Vredenburg (2003)

The model below illustrates how User Centered Design (UCD) fits into IBM's integrated product development process (IPD) and to its overall business management process.

Vredenburg noted, "Developing a new process and further enhancing it is only one component, albeit an important one, in the overall strategy of building ease of use into the total user experience at IBM. Organizations need to be enabled to carry out new processes and be provided with leadership and guidance while executing them.

UCD is a core enabling process in the overall integrated product development (IPD) process, which is the business checkpoint mechanism used for all funding and project-milestone reviews within IBM. Having UCD and UE included directly in the corporate-wide IPD process ensures that decisions made about an offering will be required to take UCD and UE information into account."

Vredenburg also noted creating new corporate-wide positions, development of education and training, communications, and collaboration programs, and provision of tools and technology as part of IBM's strategy for integrating UCD.



Sun Sigma Framework

DMADV methodology for new products

PHASES

1	2	3	4	5
Define Identify the problem or process to be fixed or to be designed.	**Measure** Identify list of customer requirements and develop and prioritize CTQs around customer expectations.	**Analyze** Customize customer expectations to create a clean paper design.	**Design** Develop one clean paper design to meet customers requirements.	**Verify** Implement the new process and verify that the design works.

ACTIVITY SUMMARY

Charter approval

SunShot (Workout) Session to identify Quick Hits

Phase and Gate approval

Update project in PRT

Customer Prioritized Needs:
Identify all customers
Prioritize customers
Assess current customer data
Collect "Voice of the Customer"
Determine "Voice of the Customer" strategy
Analyze and prioritize needs

CTO Drilldown
Determine characteristics and measures for needs
Select critical few measures
Measurement system capability
Establish targets and specs
Set performance (Sigma) goals

Benchmarking / best practice identified

Phase and Gate approval

Update project in PRT

Concept Design
ID functions required to meet Critical to Qualities (CTQs)
Develop alternate design concepts
Select most promising concepts
Develop concept design

Performance Gap
Deploy CTQs to concept design
Predict CTQ performance
Perform GAP analysis and revise concept design
Perform design risk assessment (FMEA)

Refine benefit estimates

Phase and Gate approval

Update project in PRT

Detailed Design
Develop detailed design elements
Flowdown CTQs to elements
Estimate capabilities of detailed design
Perform design risk assessment (FMEA)
Analyze gap and optimize design

Verification Plan
Customer feedback on design
Determine what to verify
Create test document
Create pilot plan

Control Plan
ID critical input, process, output parameters
Document procedures to guide ongoing operation
Exit strategy identified

Assess patentability of process/product

Phase and Gate approval

Update project in PRT

Verified Design
Build pilot product / process for verification
Execute test document
Analyze results and adjust design

Project Closure
Implement design full scale
Execute control plan
Transfer ownership

Financial benefits verified and approved

Phase and Gate approval

Update project in PRT

Celebration and recognition

OUTPUTS

Project Charter:
Business Case
Problem Statement
Goal Statement
Scope
Definition of Team and Time Commitments
Timeline / Milestones
Estimated Benefits
Risks and Constraints
Charter Approvals

High Level Process Map

Stakeholder Analysis and Actions

"Voice of the Customer" Strategy

Performance (Sigma) goals

First pass scorecard

Stakeholder analysis and actions

Alternate design concepts

Final concept design

Updated scorecard

Stakeholder analysis and actions

Detailed design elements

Test document

Pilot plan

Updated scorecard

Stakeholder analysis and actions

Pilot product / process

Updated scorecard

Final product / process

Sun Sigma Framework

DMAIC methodology for improving existing products

PHASES

1	2	3	4	5
Define Identify the problem and identify top 1 to 2 Critical to Qualities (CTQs) to focus improvements on.	**Measure** Measure the current performance of the defect (CTQ not met) and display variation in the process.	**Analyze** Analyze the process variation to determine root causes and opportunities for reducing the variation.	**Improve** Generate, select, design, test, and implement improvements.	**Control** Institutionalize the improvement and implement ongoing monitoring.

ACTIVITY SUMMARY

Charter approval

SunShot (Workout) Session to identify Quick Hits

Phase and Gate approval

Update project in PRT

Output Measurement
Project Y and defect defined
Performance specification for Y
Data collection plan
Measurement system validated
Collect data for Y
Process capability for Y
Improvement goal for Y

Benchmarking / best practice identified

Phase and gate approval

Update project in PRT

Root Cause Analysis
Brainstorm all possible Xs
Prioritize list of Xs
Verify X with data

Performance Gap
Identify gaps between capabilities and customer requirements
Re-evaluate DMAIC vs. DMADV

Benchmarking

Refine benefit estimates

Phase and Gate approval

Update project in PRT

Solution Generation and Pilot Test
Develop and select solutions
Perform pilot of solutions
Confirm improvements
Confirm prioritized Xs
Map new process
Determine new capability / performance

Implementation Plan
Develop implementation plan
Identify controls
Implement improvements

Benchmarking

Phase and Gate approval

Update project in PRT

Sustained Solution
Process documentation / controls in place
Measure performance
Confirm sustainable solution
Validate measurement system on Xs

Project Closure
Evaluation transition opportunities
Identify other improvement opportunities
Project documentation complete
Translation package

Financial benefits verified and approved

Phase and Gate approval

Update project in PRT

Celebration and recognition

OUTPUTS

Project Charter:
Business Case
Problem Statement
Goal Statement
Validated customer CTQs
Scope
Definition of team and time commitments
Timeline / milestones
Estimated benefits
Risks and constraints
Charter approval

High Level Process Map

Stakeholder Analysis and Actions

Detailed process map

Stakeholder analysis and actions

Stakeholder analysis and actions

Possible solutions

Pilot of solutions

Map of new process

Implementation plan

Final improvements

Stakeholder analysis and actions

Project documentation

Translation package

Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)

"Mapped" processes for product instances

PHASES

1	2	3	4	5	6	7	8
Concept	Plan	Develop / Integrate / Test	System Test	Customer Acceptance	Deploy	Sustain	Retire

ACTIVITY SUMMARY

1	2	3	4	5	6	7	8
Business Unit creates a Product Portfolio Steering Committee	Create optional Sub-Steering Committees Create optional Consolidation Teams (C-Teams) Influence Line Management to start projects Approve each Component Project Plan Approve each Component Project Architecture Test each component Deliver each Component Project to C-Team Conduct Component TOI Test Components Conduct Product TOI	Execute System Test Create training materials Create announcement package	Verify execution through system test of Product Plan Execute Customer Acceptance Test Announce to internal community Announce to channels Create training materials Conduct Revenue Release	Announce to the public Create and deliver Production Training Create and execute Launch Package Conduct 1st lessons learned review	Sustain the product Conduct on-going viability assessments Announce intent to end-of-life Conduct 2nd lessons learned review	Inventory cap equipment and disposables Announce end-of-life to public Execute End-of-Product Plan Conduct 3rd (final) lessons learned review

OUTPUTS

1	2	3	4	5	6	7	8
Product Requirements Document Initial Functional Specification Product Content Document	Functional Specification Product Plan Document Initial Product Contents List (BOM) 1-pagers for required components	Component Plans Internal Design Specification Component & Integration Test Plans 1-pagers for components Finalized PPD Ops/Mrg Plan	Finalized Sales Plan System Test Reports Updated Product Boundary & Summary Approved Product Content List (BOM) Defect Resolution and Risk Assessment Reports Finalized PPD Marketing Plan Training Materials Announcement Package	Finalized Support Plan Customer Acceptance Test Report Defect Resolution and Risk Assessment Reports Training Materials		End-of-Product Plan	Finalized End-of-Support-Life Plan

Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)

"Mapped" processes for product lines

PHASES

1	2	3	4	5	6	7	8
Concept	Plan	Develop / Integrate / Test	System Test	Customer Acceptance	Deploy	Sustain	Retire

ACTIVITY SUMMARY

BU creates Product Line Portfolio Steering Committee (PC)	Develop the product and verify component performance Develop Sun's and critical partners' production processes Prepare comprehensive plans for introduction and support	Qualify the product functionality and performance Qualify production and supplier processes	Confirm product functionality in planned customer environments Execute introduction and support plans	Launch the product Stabilize operational and support processes	Manage product profitability and growth Provide customer support and defect tracking	Retire product and manage customer transitions

OUTPUTS

Product Line Requirements Document Initial Functional Specification Product Line Content Document	Functional Specification Product Line Plan Document

Complex linear models

Most of models of the design process have three to seven steps. If they contain more steps, they're typically organized into a tree with three to seven major steps.

This may be another function of George Miller's famous "Magic Number 7."

The next section includes some very detailed models.

Vanguard Group

The model on the following spread comes from the design team at the Vanguard Group. So far as I know, they have not published it.

Web development process
after Vanguard Group (circa 1999)

Major review point.
Team must receive Senior Management approval to proceed.

PHASES: Planning		Requirements & Conceptual Design				Analysis & Design
1	2	3	4	5	6	7
Idea ► Create Business Case and Initial Scope	Complete Initial User Research	Create Site Architecture and Conceptual Design	Begin Functional Requirements	Create Navigational Diagrams	Complete Conceptual Testing	Create Detailed Wireframes
ACTIVITY SUMMARY						
Conduct user research Develop business case summary Conduct competitive analysis Define goals and objectives	Develop user research plan Create user profiles Begin task analysis Begin to establish mental model	Refine profile mapping to segment strategy Refine task analysis Refine mental model Create outline of containers and their relationships Begin to "thumbnail" high-level look and feel Validation of concepts with the business	Heavy iteration of requirements, architecture and design	Build architecture from container / relationship outline and mental models Identify detailed content inventory Map detailed content to architecture Conduct gap analysis	Identify and prioritize key content Identify top tasks for testing Validate requirements and architecture with users Prepare appropriate materials for test Conduct tests Create recommendations based on findings	Create "visualization" of requirements Audit pages to uncover functional and non-functional requirements Begin to refine & apply design standards
OUTPUTS						
Business case	Client Interviews Report of findings	Site architecture & design concepts	Requirements	Navigational diagrams	Paper mock-ups Usability report	Wireframes (iterated)
VALUE ADD *(with lessons learned)*						
Web team participation at brainstorming stage and during strategy formulation	Separate user goals & business strategy, and then reconcile discrepancies Involve the right people early in the planning process	Control complexity Understand user goals and behaviors to avoid low usage Cohesive team environment creates efficient, informed output and rapid iteration	Gain business buy-in before entering into costly design development cycle Reduces complexity (tasks, language, metaphors, etc.) Creates clarity around tasks, focuses the effort Assists with business prioritization (real estate) High-level technology assessment	Balance business objectives and user needs/expectations Asses design standards and determine if differentiation is necessary and why Translation of competitive analysis to concept	Keep Important tasks within a few clicks Map architecture back to business strategy and user goals Consider usage reporting Clarify scope and functionality Consider cross-over experience	Usability expertise allows for rapid validation & adjustment of architecture & requirements Business analysis of test recommendations Judgement in applying the right usability techniques to the project	Ensure that design is scaleable and flexible to allow for ongoing maintenance Allow business to see and participate in the evolution of the site Consider error conditions Additional design strategy / standards analysis

Major review point.
Team must receive Senior Management approval to proceed.

Refinement of UI Design

Build / Monitor / Improve

8	9	10	11	12		
Complete User Testing of Wireframes	Create Design Strategy & Design Guideline Specification	Complete User Interface	Complete Testing of Final User Interface	Deliver UI Specifications and Related Documentation	Build / Elevate / Test	Continuous Improvement
Prepare next level of tasks and test materials Conduct test Report usability test findings Interpret findings and make recommendations	Assemble the design strategy and document Refine and apply site specific design guidelines Identify key content & place Validate design strategy with the business	Create detailed design for all unique instances	Conduct final testing to uncover major flaws Focus testing on unique instances Create recommendations based on findings	Finalize UI specifications & prototype of unique instances Deliver and review UI with the development team		Gather data Uncover issues and defects Use DMAIC to identify improvements Work with business to submit SCRs and project requests Follow SDLC through to elevation Monitor improvements through data
Usability report	Design strategy	"Skinned prototype"	"Skinned prototype" (iterated) Usability report	Prototype & documentation		Date tracking an analysis VOC Paretos Web Usage Reports Error Logs Exit Surveys WTSS Trends NICE calls
Predict user interest Incorporating multiple options into tests to reduce complexity for the user Business analysis of test recommendations Implement / adhere to change control process Look for compatibility across browsers, OS, and downloads, etc.	Look and feel of the site is being established and validated with key business stakeholders Evaluation of page loads Manage business feedback by area of expertise (avoid design becoming the focus versus content)	High degree of design / business iteration Create a site that "hangs together" Covering all possibilities through unique instances	Final checkpoint with users to validate the experience (design, architecture, content, and usability) Finalize template for usage reporting	Continual improvement of documentation to create efficiencies between design and development	Thorough attention to integration test Quality and control assessment by design team in the integration region Page performance analysis and trouble-shooting	Detailed analysis of all data (PCE, VOC, WUR, WISS, PVCS Tracker, Research, etc.) VOC analysis VUE projects Prioritization of enhancements with the business

Alan Cooper

Few people are good computer programmers and also good interaction designers. Alan Cooper is one. Cooper's favorite topic is what's wrong with the software that increasingly fills our lives and how it came to be so bad. He holds forth on the subject in two books, *About Face: The Essentials of User Interface Design* (1995) and *The Inmates Are Running the Asylum: Why High-Tech Products Drive Us Crazy and How to Restore the Sanity* (1999).

In summary, Cooper's argument is as follows: In software, the cost of adding one more new feature is almost nothing; no additional material or manufacturing costs restrain feature creep. The trouble is: Each additional feature makes the product more complicated to understand and more difficult to use.

In the traditional software development process, many people inside a company—and oftentimes customers as well—ask for features. Thus, a list of features often becomes the de facto product plan. Programmers make this approach worse by picking or negotiating their way through the list, often trading features for time. In such a process, Cooper points out, it's hard to know when a product is complete.

Cooper advocates five significant changes to conventional methods of software development in his goal-directed design process:

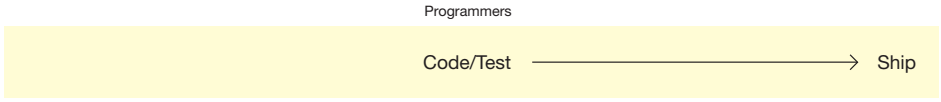
1) Design first, program second.
2) Separate responsibility for design from responsibility for programming.
3) Hold designers responsible for product quality and user satisfaction.
4) Invent on specific user for your product—a persona. Give that user a name and an environment and derive his or her goals.
5) Work in teams of two: designer and design communicator

I developed the diagrams that follow based on a series of conversations with Cooper and members of his staff, including Dave Cronin, David Fore, Kim Goodwin, Jonathan Korman, and Robert Reimann. The first two were first published in the AIGA journal, *Gain*. The second two have not been published previously.

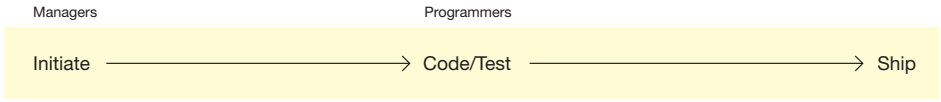
Evolution of the software development process
after Alan Cooper (2001)

In 1975, Cooper borrowed $10,000 from his dad and started a company with his high-school friend, Keith Parsons. They began writing and selling software for personal computers. The diagram below describes the evolution of the software development process from the beginning of the personal computer industry to the present, as Cooper saw it.

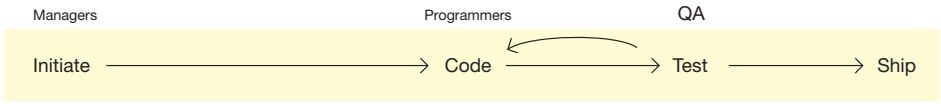
Originally, programmers did it all:
In the early days of the PC software industry, smart programmers dreamed up useful software, wrote it, and even tested it on their own. As their businesses grew, the businesses and the programs became more complicated.



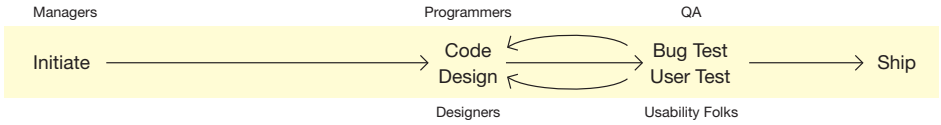
Managers brought order:
Inevitably, professional managers were brought in. Good product managers understand the market and competitors. They define software products by creating requirements documents. Often, however, requirements are little more than a list of features, and managers find themselves having to give up features in order to meet schedules.



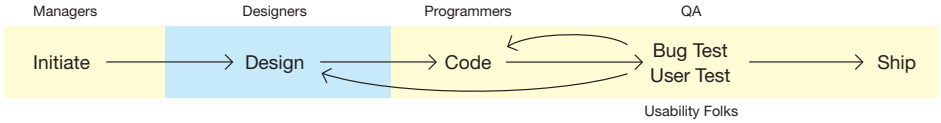
Testing became a separate step:
As the industry has matured, testing has become a separate discipline and a separate step in the process. Today, it's common to find 1 tester for every 3 or 4 programmers. This change illustrates that the programmer's role is not fixed but still evolving.



Today, common practice is to code and design simultaneously:
In the move from command-line to graphical user interface, designers became involved in the process — though often only at the end. Today, common practice is for simultaneous coding and design followed by bug and user testing and then revision.



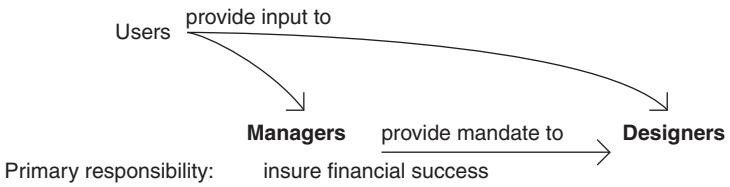
Cooper insists that design precede programming:
In Cooper's goal-directed approach to software development, all decisions proceed from a formal definition of the user and his or her goals. Definition of the user and user goals is the responsibility of the designer — thus design precedes programming.



Goal-directed design process
after Alan Cooper (2001)

Cooper puts user goals at the center of the software design process. That process is part of a series of office practices which depend on the talent and skills of designers and on their application of principles and patterns throughout the process.

This diagram shows the process proceeding in steps from left to right. It leaves out feedback loops and iteration which are necessary for producing good work.



Initiate	Design

Research and Analyze Opportunities, Constraints, and Context
(focus in the first half, continuing throughout)
Who will use the product?
What problem will it solve for them?

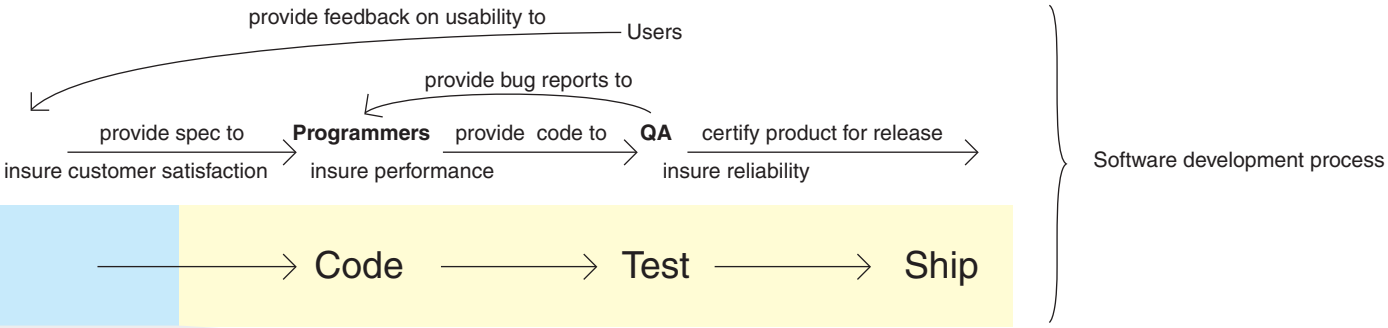
Activity:	Define intent and constraints of project	Review what exists (e.g. documents)	Discuss values, issues, expectations	Apply ethnographic research techniques	Define typical users	Deduce what users want
Result:	**Scope** desired outcomes time constraints financial constraints general process milestones (Scope may be loose or tight.)	**Audit** business plan marketing plan branding strategy market research product plan competitors related technology	**Interviews** management domain experts customers partners sales channel (This step leads to a project mandate.)	**Observations** use patterns potential users their activities their environments their interactions their objects (tools) (aeiou framework from Rick Robinson, Sapient)	**Personas** primary secondary supplemental negative served (indirectly) partner customer organizational	lead to → **Goals** life end experience personal practical corporate false
Artifact:	Project Brief	Summary Insights	Tapes Transcripts Summary Insights	Tapes Transcripts Summary Insights	Notes	Notes
Meetings:	Briefing	-	Interviews	Chalk talk (early findings)	-	Chalk talk with management

Design Office Practices

The way the office is set up and run – the environment, the spoken and unspoken rules – affect the work. Cooper's staff describes several key practices:
- goal-directed design process
- collaborative environment and common purpose
- D/DC team structure (see separate diagram)
- egoless design
- appropriateness of assignments
- commitment to education
- commitment to enhance process
- assessment and self-assessment

Designer Talent and Skills

A designer's native abilities and background also affect the work. Cooper looks for people with these skills:
- analytic
- conceptual
- visual
- written
- communications
- empathic
- interpersonal
- brainstorming
- imagination



The goal-directed design process takes place within a larger software development process.

Synthesize and Refine

(ongoing throughout, focus in the second half)

Form, Meaning, and Behavior

What is it?
How will it behave for users?

	Concept	Scenarios	Elements	Framework	Spec
	Imagine a system to help users reach goals	Tell stories about using the system	Derive components based on users	Organize the components	Refine details; describe models
drive*	problem definition vision definition design imperatives (May require changes in scope.)	day-in-the-life key-path error set-up	information objects functional objects control mechanisms	object relationships conceptual groupings patterns logic / narrative flow navigation structure	appearance language flow / behavior product character product story
	Formal Document Problem Statement Vision Statement	Notes Storyboards	Lists Sketches Diagrams High-level data models	Sketches Flow Diagrams	Formal Document Demonstration Prototype
	Presentation	-	-	Chalk talk with programmers	Presentation

* spark
inform
motivate
filter
organize
prioritize
inflect
validate

Throughout the goal-directed design process, designers apply other practices, their talent and skills, as well as principles and patterns.

Design Principles

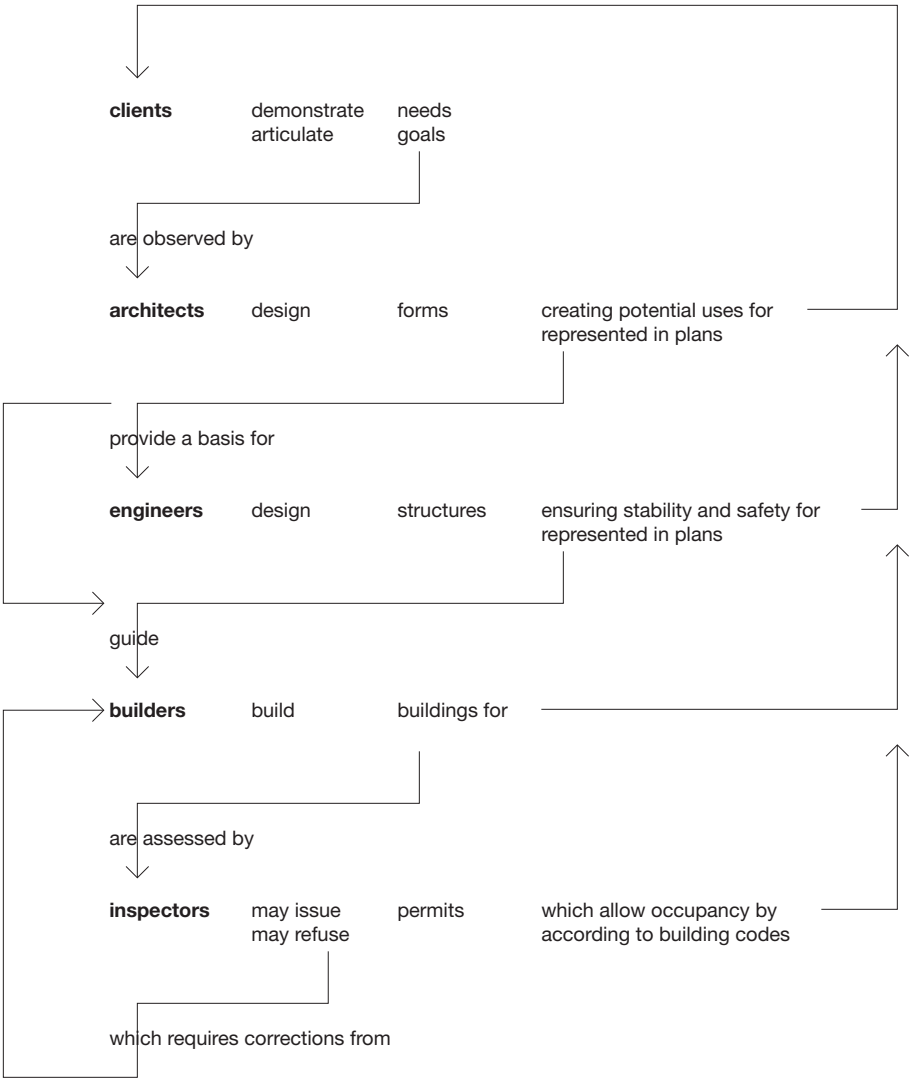
Principles guide the choices designers make as they create. Principles apply at all levels of design from broad concept to small detail. For example:
- Do no harm. (Hippocrates)
- Meet user goals.
- Create the simplest complete solution. (Ockham, Fuller)
- Create viable and feasible systems.

Design Patterns

Design patterns are recurring forms or structures which designers may recognize or apply – during analysis and especially during synthesis. Christopher Alexander, "A Pattern Language," provides examples of patterns for architecture; Cooper collects patterns for software interaction. For example, a common pattern is dividing a window into two panes: the left smaller pane provides tools or context and the right larger one provides a working space or details.

Idealized process of developing buildings
after Alan Cooper (2004)

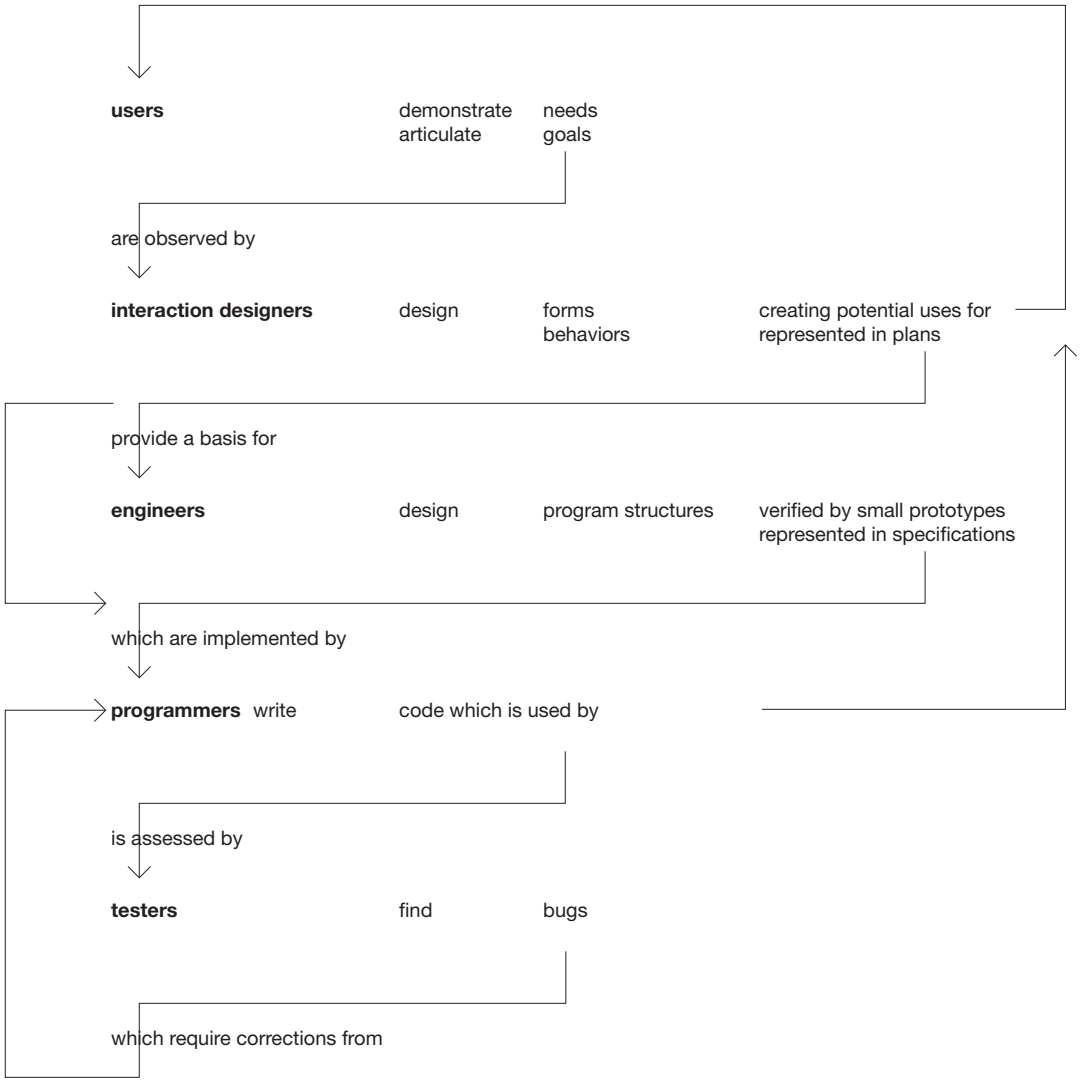
Since high school, architecture has fascinated Cooper. His view of how architecture should be practiced provides a model for how he believes software development should be practiced. Cooper organized the process of developing a building into three domains: architecture, engineering, and construction. In his view, architects determine what the building will be like (how it will "behave"). Based on the architect's plans, engineers determine how to make the building stand up. And finally, the builders execute the architect's and engineer's plans. Obviously, architects serve their clients and consult with engineers and builders on what is possible and practical.



Idealized process of developing software
After Alan Cooper (2004)

Following his ideal model of architecture, Cooper advocated organizing the process of developing software into three domains: interaction design, engineering, and programming. Interaction designers determine what the software will be like (how it will "behave"). Based on the interaction designer's plans, engineers determine how to make the software work by writing many very short test programs—but no final code. And finally, the programmers write real code to execute the interaction designer's and engineer's plans. Here too, Cooper acknowledges the need for feedback—for interaction designers to observe users to understand their goals, to consult with engineers to understand what's possible, and finally to consult with programmers to answer questions as they program.

Cooper distinguishes engineers from programmers. According to him, engineers like to figure out how to solve problems. They like to create and don't want to be told what to do. Programmers, he suggested, don't like ambiguity. They like to code and simply want to know what the code is suppose to do. Cooper warned, putting an engineer in a programming job or a programmer in an engineering job is a recipe for unhappiness.



Morris Asimow

Asimow defines morphology of design as “the study of the chronological structure of design projects.” He notes, “Each design-project has an individual history which is peculiarly its own. Nonetheless, as a project is initiated and developed, a sequence of events unfolds in a chronological order forming a pattern which, by and large, is common to all projects.” He continues, “Design is a progression from the abstract to the concrete. (This gives a vertical structure to a design project.) . . . Design is [also] an iterative problem-solving process. (This gives a horizontal structure to each design step.)”

Asimow defines the phases of a project (vertical) as
- Feasibility study
- Preliminary design
- Detailed design
- Planning for production
- Planning for distribution
- Planning for consumption
- Planning for retirement

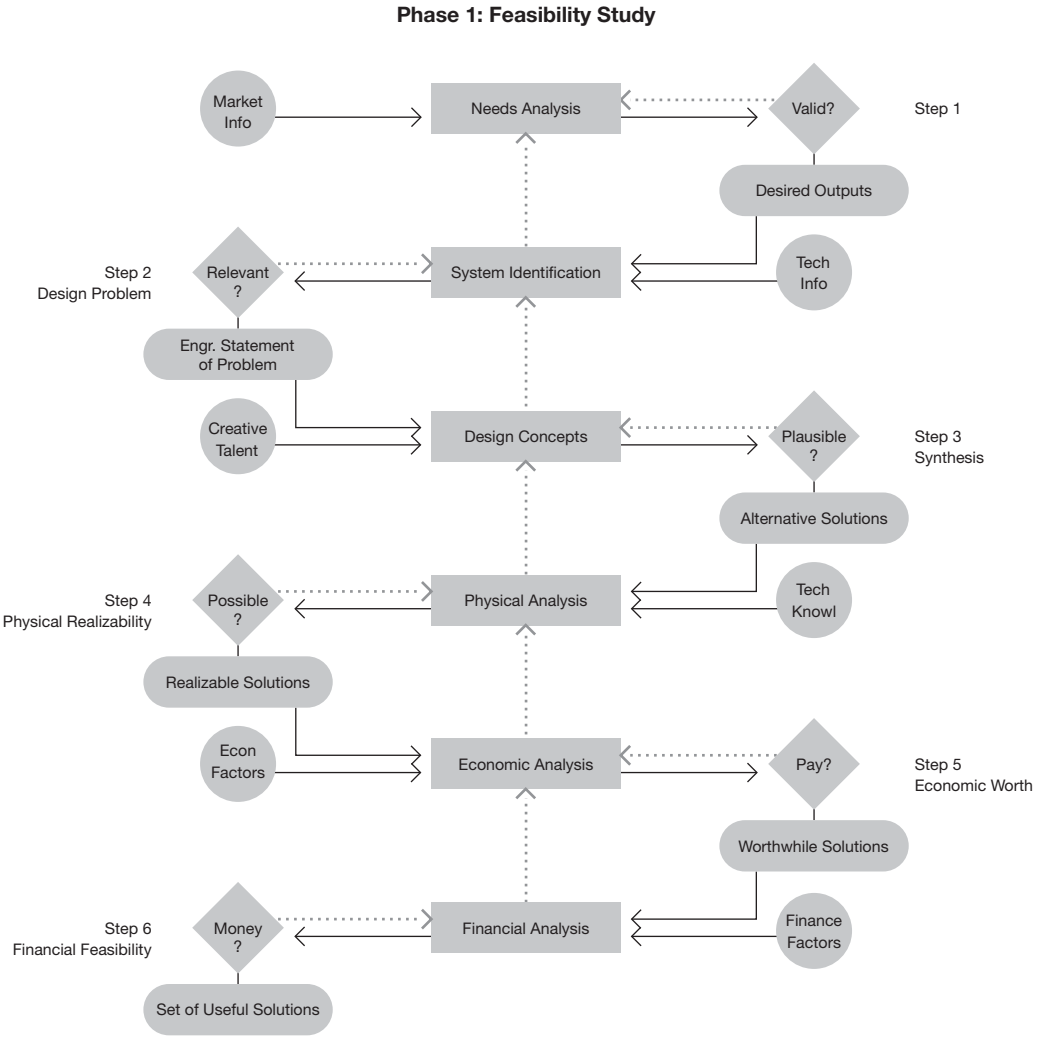
He likened the design process (horizontal) to “the general problem solving process,” describing these steps”
- analysis
- synthesis
- evaluation
- decision
- optimization
- revision
- implementation

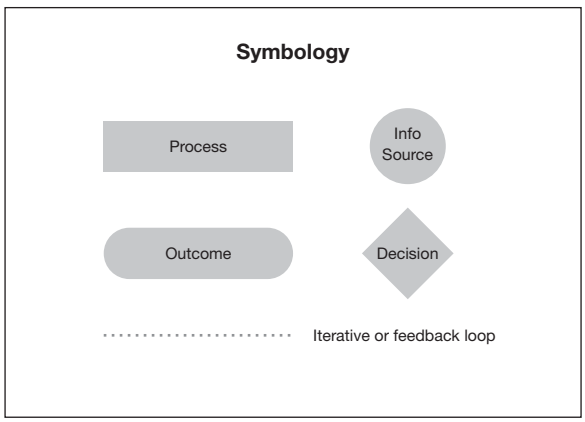
In *Introduction to Design*, Asimow devotes more than 50 pages to describing engineering design and the design process. He defines engineering design as “a purposeful activity directed toward the goal of fulfilling human needs, particularly those which can be met by the technological factors of our culture. . . . As a profession, Engineering is largely concerned with design. What distinguishes the objects of engineering design from those of other design activities is the extent to which technological factors must contribute to their achievement.”

Asimow, like Alexander, Jones, and Doblin, distinguishes craft-based design, “design by evolution,” from “design by innovation.” He notes, “Now more frequently than ever in the past, products are designed *de novo*,” and suggests this creates greater risk and complexity and thus implies the need for new design tools (the subject of his book.)

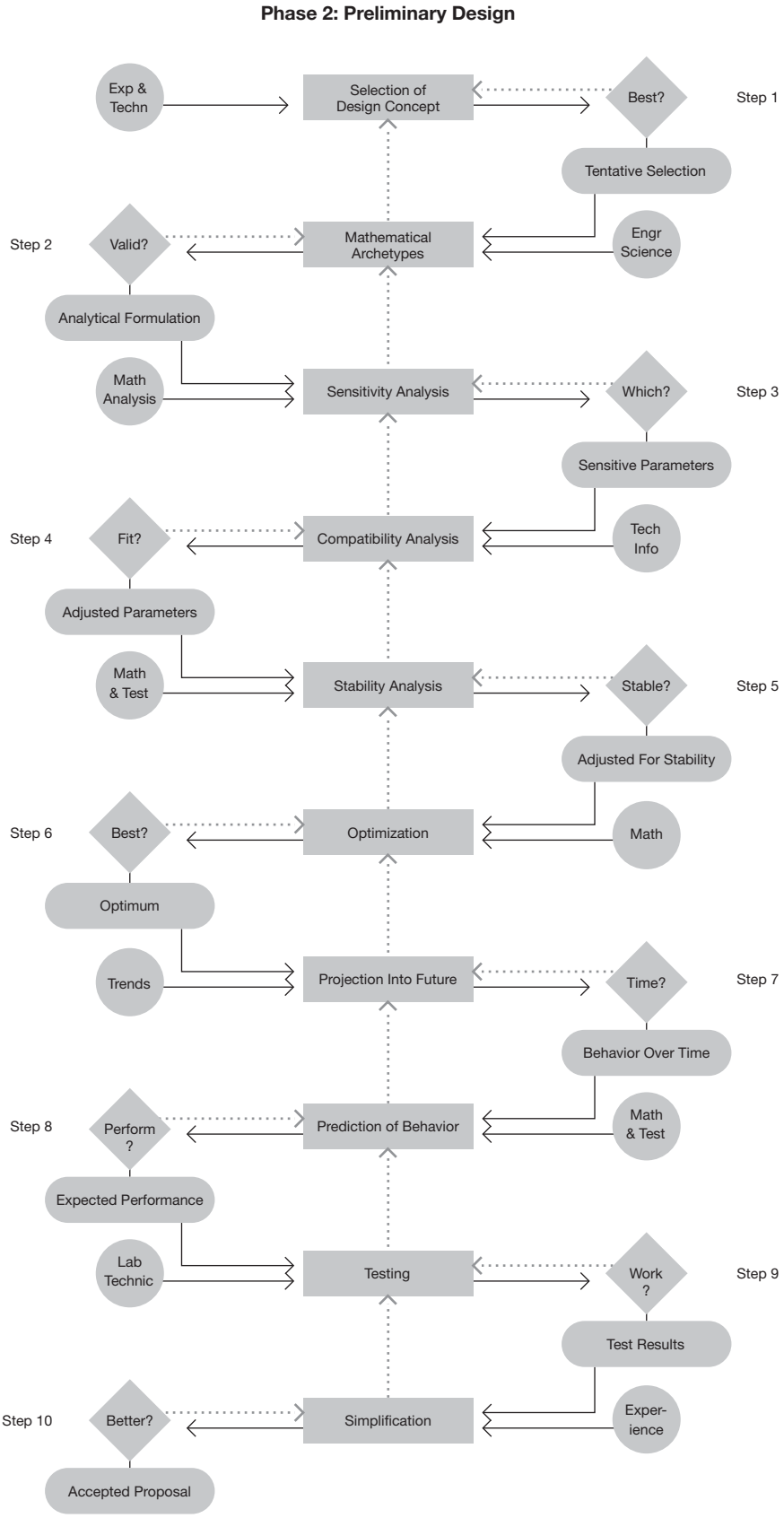
According to Rowe (1987), Asimow was “an industrial engineer prominent in the 1950s and 1960s.” Two years after Asimow first published his model, Tomas Maldonado and Gui Bonsiepe introduced it to the design and architecture community, including it in their seminal article “Science and Design” published in the journal, *Ulm* 10/11 (1964).

Morphology of design (1 of 3)
after Morris Asimow (1962)

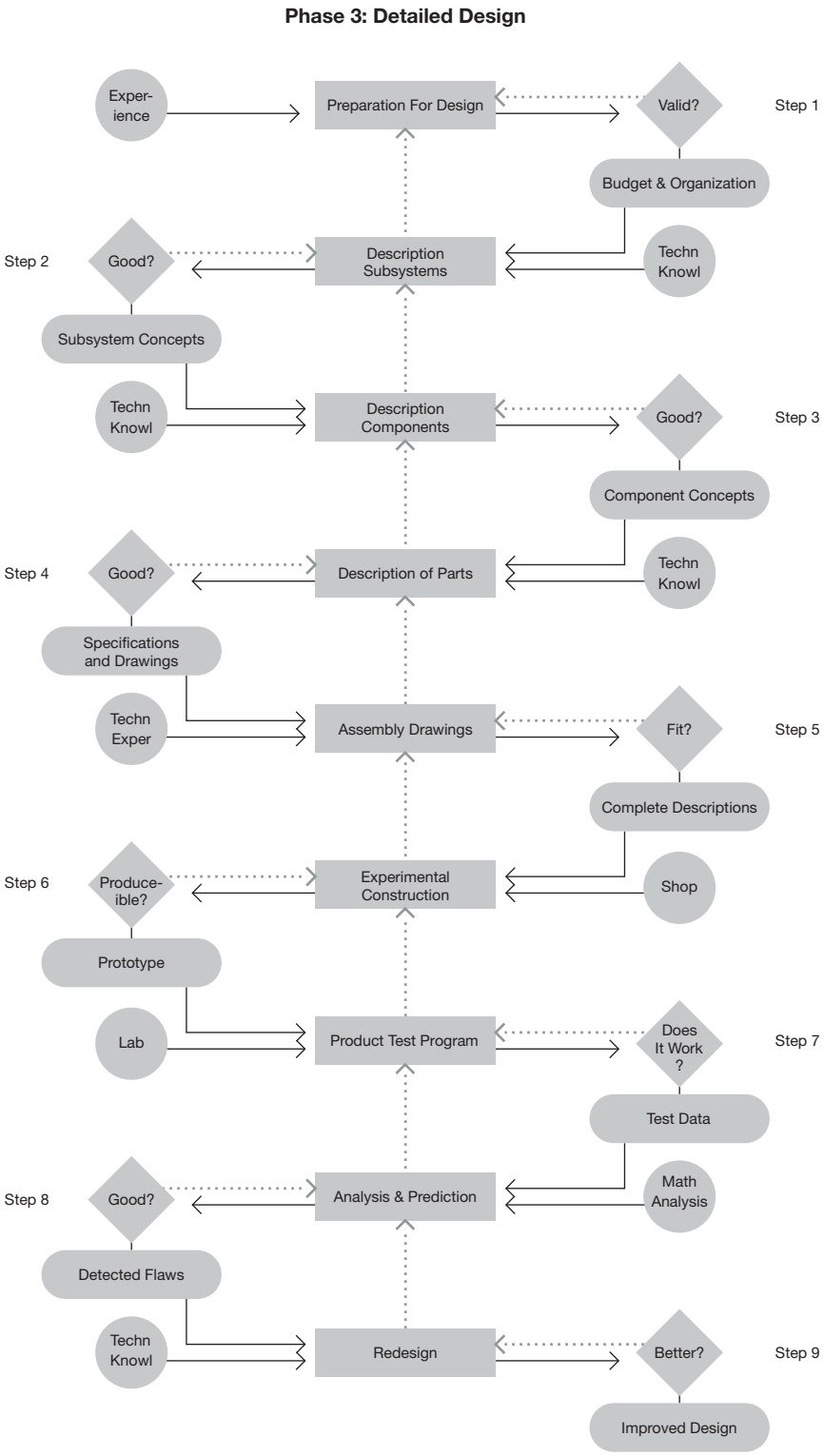




Morphology of design (2 of 3)
after Morris Asimow (1962)



Morphology of design (2 of 3)
after Morris Asimow (1962)



Bruce Archer

Cross (1984) notes, “One of the first tasks attempted by the design methodologists was the development of new, systematic design procedures.” He calls out four authors as especially important: Jones, Alexander, Archer, and Rittel. (For more on Jones, see pages 56-59; for more on Alexander, see page 18.) This section presents three models from Archer. (Rittel came to see design as a process of argumentation aimed at coming to agreement on goals; as far as I know, he presented no staged or procedural models of design.)

Archer taught at both the Royal College of Art (RCA) in London and the Hochschule für Gestaltung in Ulm (HfG Ulm). Rowe (1987) notes that at Ulm, “speculation moved beyond description and explanation of design behavior and into the realm of idealization. Not only was the possibility of a ‘scientific’ and totally objective approach toward design seriously entertained, it became a goal in itself. A confident sense of rational determinism prevailed; the whole process of design, it was believed, could be clearly and explicitly stated, relevant data gathered, parameters established, and an ideal artifact produced.”

Archer’s statements about the design process contradict Rowe’s critique, “The fact is that being systematic is not necessarily synonymous with being automated.” Archer continues, “When all has been said and done about defining design problems and analyzing design data, there still remains the real crux of the act of designing—the creative leap from pondering the question to finding a solution. . . . If we accept that value judgments cannot be the same for all people, for all places or all time, then it follows that neither the designer nor his client (nor, eventually, the user) can abdicate the responsibility for setting up his own standards. Similarly, there is no escape for the designer from the task of getting his own creative ideas. After all, if the solution to a problem arises automatically and inevitably from the interaction of the data, then the problem is not, by definition, a design problem.”

Biological sequence of problem solving
after Bruce Archer (1963-1964)

Archer notes the similarity of biological response mechanisms and problem solving in computer programming and design. And he explicitly links these processes to cybernetics.

"The study of control mechanisms of living organisms is called cybernetics. In recent times, designers of highly complicated control systems for machine tools, aeroplanes, rockets and remote controlled instruments have turned to cybernetics for inspiration."

"A further line of thinking which does not quite fall into this pattern but which has contributed to the development of systematic methods for designers is the 'heuristic'. an ancient philosophical study of the method of intellectual discovery which has been revitalized recently by Professor [George] Polya of Stanford University, USA."

"The method for solving design problems set out in this article owes something to both the heuristic and the cybernetic approaches." (See Polya's model on page 36. See models from cybernetics on pages 117-118.)

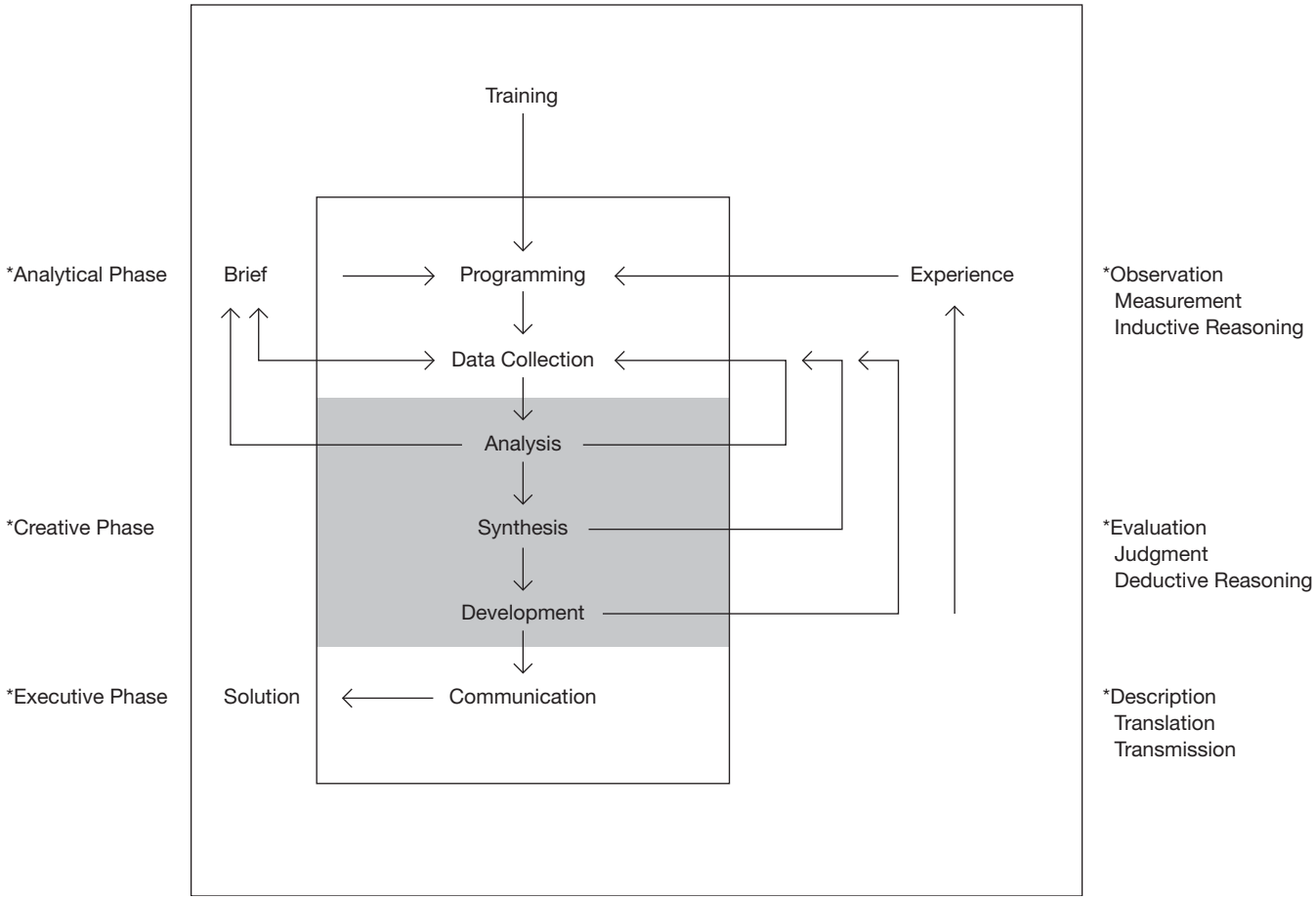
01 Discern 'something wrong'
02 Heighten alertness and locate the area of disturbance
03 Bring appropriate sense organs to bear
04 Evaluate evidence and compare with previous experiences
05 Postulate a possible cause for the disturbance
06 Recall experience with similar and/or analogous causes
07 Predict consequence of suggested cause
08 Formulate courses of action possible in response to such a cause
09 Recall experience with similar and/or analogous course of action
10 Predict consequences of each suggested course of action
11 Select course of action to be followed
12 Act
13 Discern effect of action
14 Recall experience with similar and/or analogous effects
15 Repeat from 07 until equilibrium is restored

Basic design procedure
after Bruce Archer (1963-1964)

This diagram was reprinted in the journal *Ulm* (1964) and several other places, e.g., Cross (1984, 2000) and Rowe (1987). Archer proposed this model as representative of an emerging "common ground" within the "science of design method" even while acknowledging continuing "differences".

Regarding the procedure, he points out, "In practice, the stages are overlapping and often confused, with frequent returns to early stages when difficulties are encountered and obscurities found."

He continues, "The practice of design is thus a very complicated business, involving contrasting skills and a wide field of disciplines. It has always required an odd kind of hybrid to carry it out successfully. The more sophisticated the demands of function and marketing become, the harder the job of the designer will get. Already it has become too complicated for the designer to be able to hold all the factors in his mind at once."



Check List for Product Designers

Forward & Summary
The check list which accompanied the original series of articles in *Design* was regarded by the author as the embodiment of a hypothesis on the structure of the design act. Since this was published, further fundamental study has been undertaken.

The following check list is thus presented as a second hypothesis, which the author recognizes as still naive in places. It is not intended to be read as a narrative. Nevertheless, study of the summary below might be useful in presenting an overall picture of design procedure. The remainder of the check list is offered as a design tool, calculated to be useful in the control of most normal product design projects.

How to use check list and arrow diagrams
The check list has been set out in the form of a list of activities and events according to the conventions of network analysis.

It is suggested that the diagram appropriate to the phase which has been reached in the design project in question should be mounted on the wall adjacent to the designer's drawing board. As the work progresses, the designer should identify events in the check list as they take place, and tick them off on the diagram. The links in the diagram show what must be done next. Target dates and/or estimated working hours can be added where appropriate.

Phase 0

Preliminaries

Receive Enquiry
Evaluate Enquiry
Estimate Office Work Load
Prepare Preliminary Response

Phase 1 – Receive brief, analyze problem, prepare detailed programme and estimate.

Briefing

Receive Instructions
Define Goals
Define Constraints

Programming

Establish Crucial Issues
Propose A Course Of Action

Phase 2 – Collect data, prepare performance (or design) specification, reappraise, proposed programme and estimate.

Data Collection

Receive Instructions
Collect Readily Available Info.
Classify And Store Data

Analysis

Identify Sub-problems
Analyze Sub-problems About Ends
Prepare Performance Specification
Reappraise Program And Estimate

Phase 3 – Prepare outline design proposal(s).

Synthesis

Receive Instructions
Resolve Remaining Problems About Ends
Postulate Means For Reconciling Divergent Desiderata In Performance Specification
Develop Solutions In Principle to Problems About Means Arising From Performance Specification
Postulate Outline Overall Solution(s)

Phase 4 – Develop prototype design(s).

Development

Receive Instructions
Define Design Idea
Erect A Key Model
Develop Sub-problem Mutual Solution
Develop Overall Solution(s)

Phase 5 – Prepare (and execute) validation studies.

Development (Continued)

Validate Hypotheses

Phase 6 – Prepare manufacturing documentation.

Communication

Define Communication Needs
Select Communication Medium
Prepare Communication

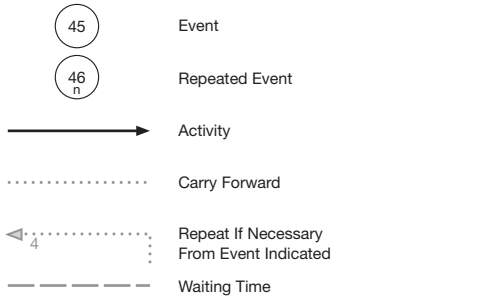
Winding Up

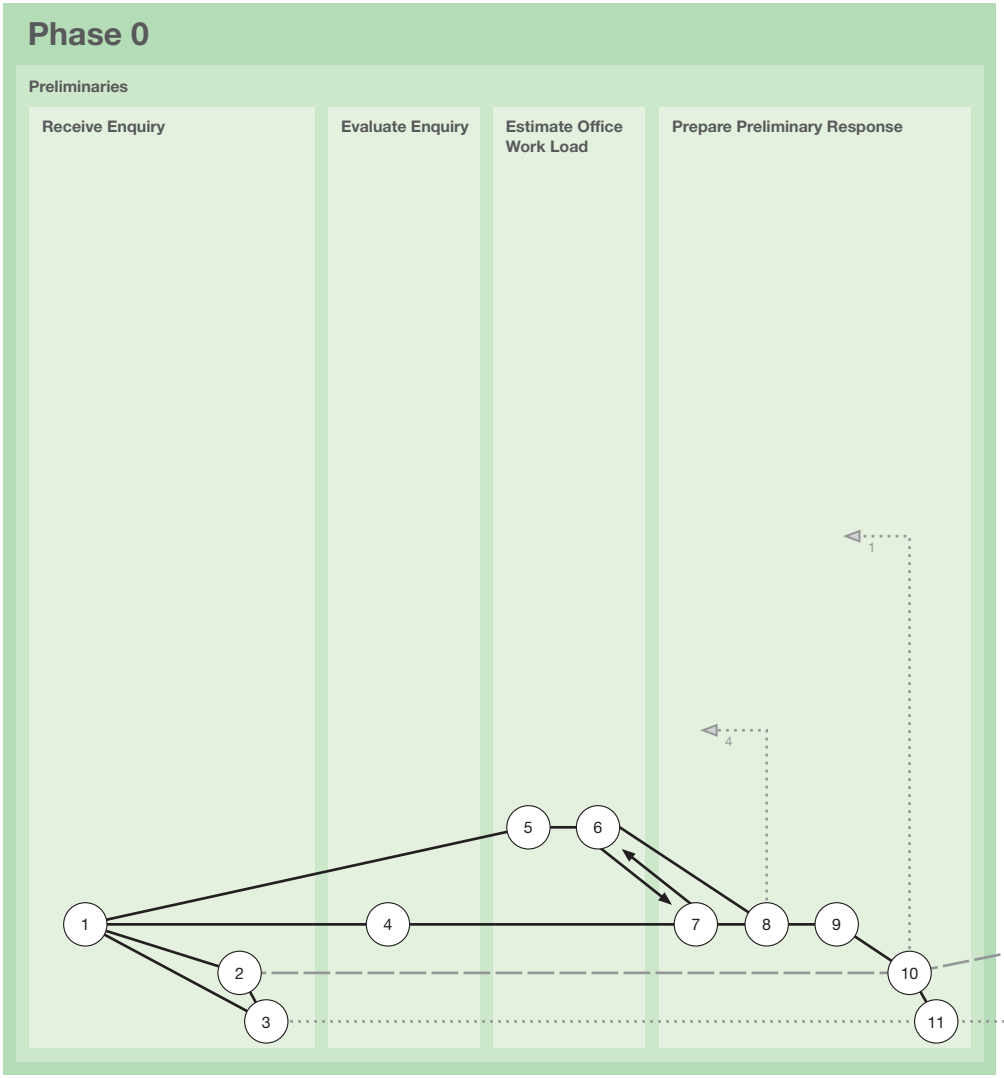
Wind Up Project
Close Records

Explanation
In the diagrams, time flows from left to right across the page, each arrow represents an on-going activity, which takes a greater or lesser amount of time. However, there is not attempt to make the length of the arrow proportional to the time taken by the activity. The circles at each end of the arrow represent the events which open and close an activity. Events occur at an instant in time, rather than over a period of time. Sometimes more than one activity is required to complete an event. Sometimes an event permits more than one activity to commence. Events with the suffix "n" must be repeated a number of times for different sub-problems.

The conventions described are common to various forms of network analysis, for example critical path planning and PERT.

Key

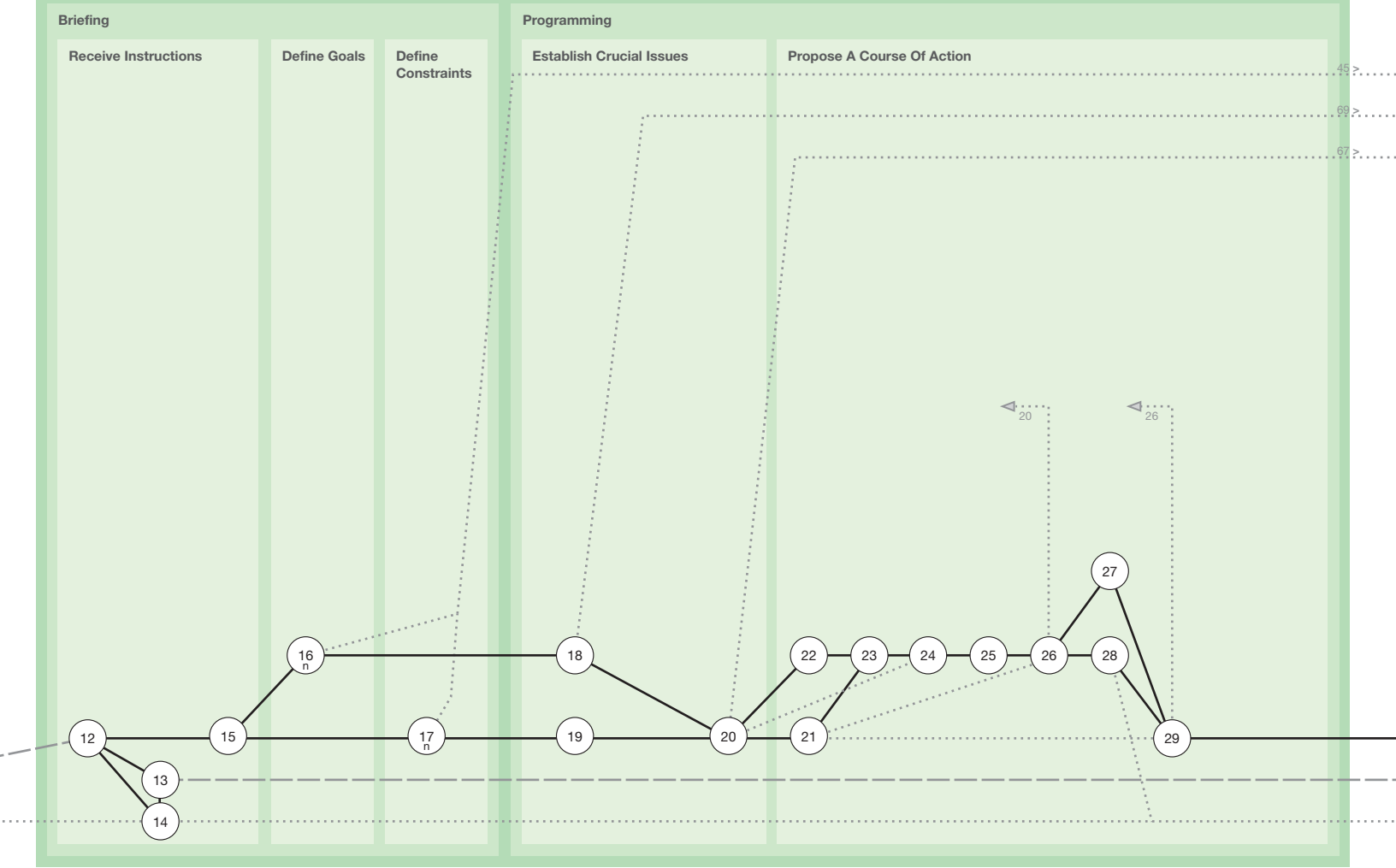




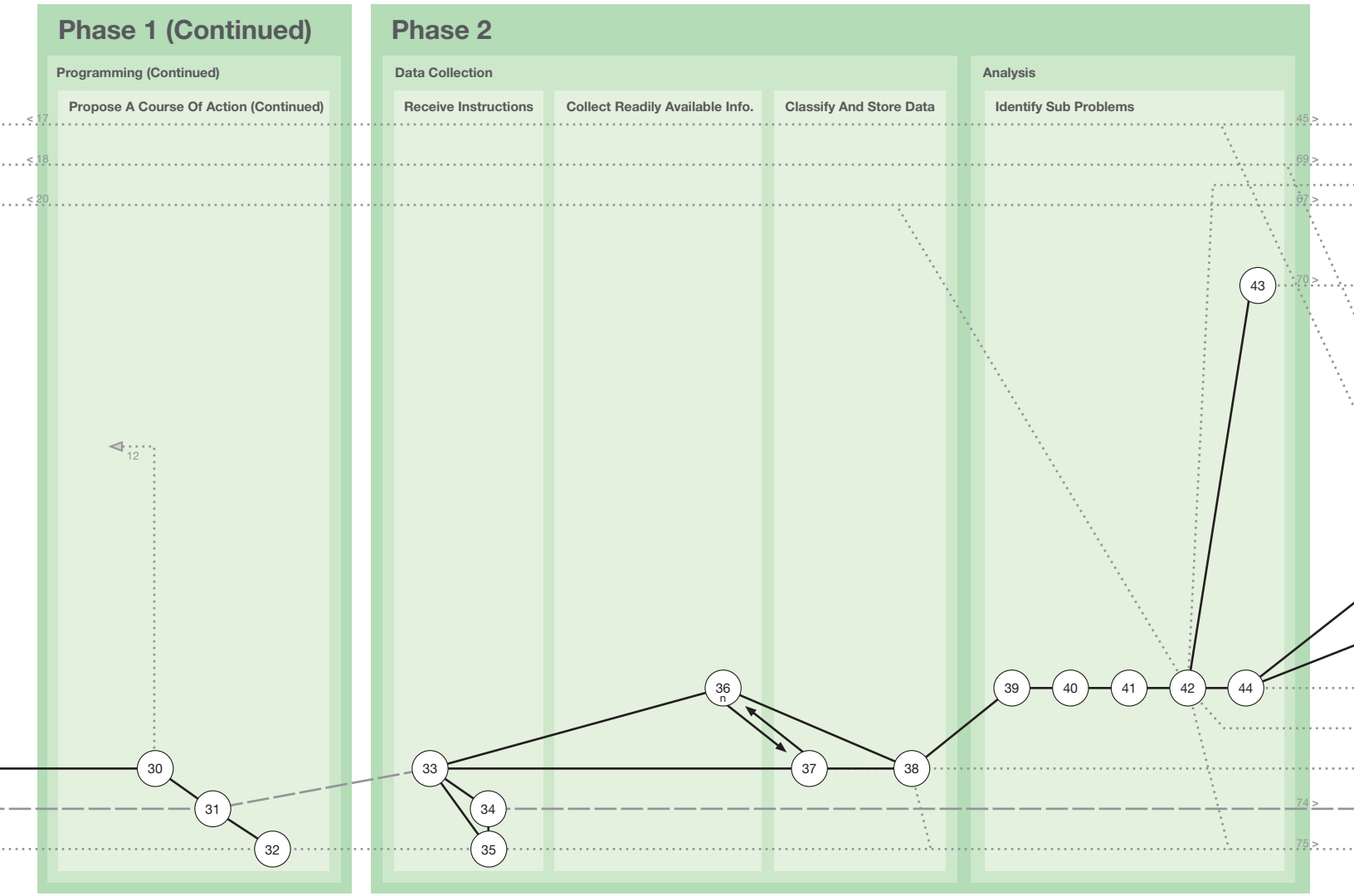
Item	Activity	Activity No	Event	
0	**Preliminaries**			
0.1	**Receive Enquiry**	1-2	1	Enquiry Received
0.1.1	Send Acknowledgment	1-3	2	Acknowledgment Dispatched
0.1.2	Open Project File	2-3	3	Project File And Progress Machinery Ready
0.1.3	Commence Chart For Recording Progress			
0.2	**Evaluate Enquiry (For Example):**	1-4	4	Report On Evaluation Of Enquiry Ready
0.2.1	Identify Authority To Whom Answerable			
0.2.2	Identify Type Of Task			
0.2.3	Identify Class of Product			
0.2.4	Define Form Of Submission Required			
0.2.5	Define Any Facility Or Free Structure Offered			
0.2.6	Define Any Program Limitations Imposed			
0.3	**Estimate Office Work Load**	1-5	5	Survey Of Office Workload Ready
0.3.1	Survey Existing And Projected Programmed	5-6	6	Estimate Of Manpower Availability Ready
0.3.2	Estimate Manpower Availability			
0.4	**Prepare Preliminary Response**			
0.4.1	Break Down Task Described Under 0.2 Into An Outline Program	4-7	7	Outline Project Program Ready
0.4.2	Match Outline Program With Manpower Availability Estimate Set Out Under 0.3.2	6-7		
0.4.3	Prepare Preliminary Estimate Of Costs	6-8		
0.4.4	Formulate A Draft Proposal	7-8	8	Draft Proposal And Estimate Ready
0.4.5	Check Draft Proposal Estimate	8-9	9	Draft Proposal And Estimate Approved
	If Necessary, Reiterate From 0.4.1 Until A Satisfactory Proposal Is Drafted			
0.4.6	Prepare And Dispatch Fair Copy Of Proposal And Estimate	2, 9-10	10	Proposal And Estimate Dispatched
0.4.7	Bring File And Progress Machinery Up-to-date	3, 10-11	11	Files And Progress Machinery Up To Date

Reiterate Section 0 Until Agreement Is Achieved On A Commission To Carry Out Phase 1 - "Receive Brief, Analyze Problem, Prepare Detailed Program And Estimate", Or Until Project Is Abandoned

Phase 1



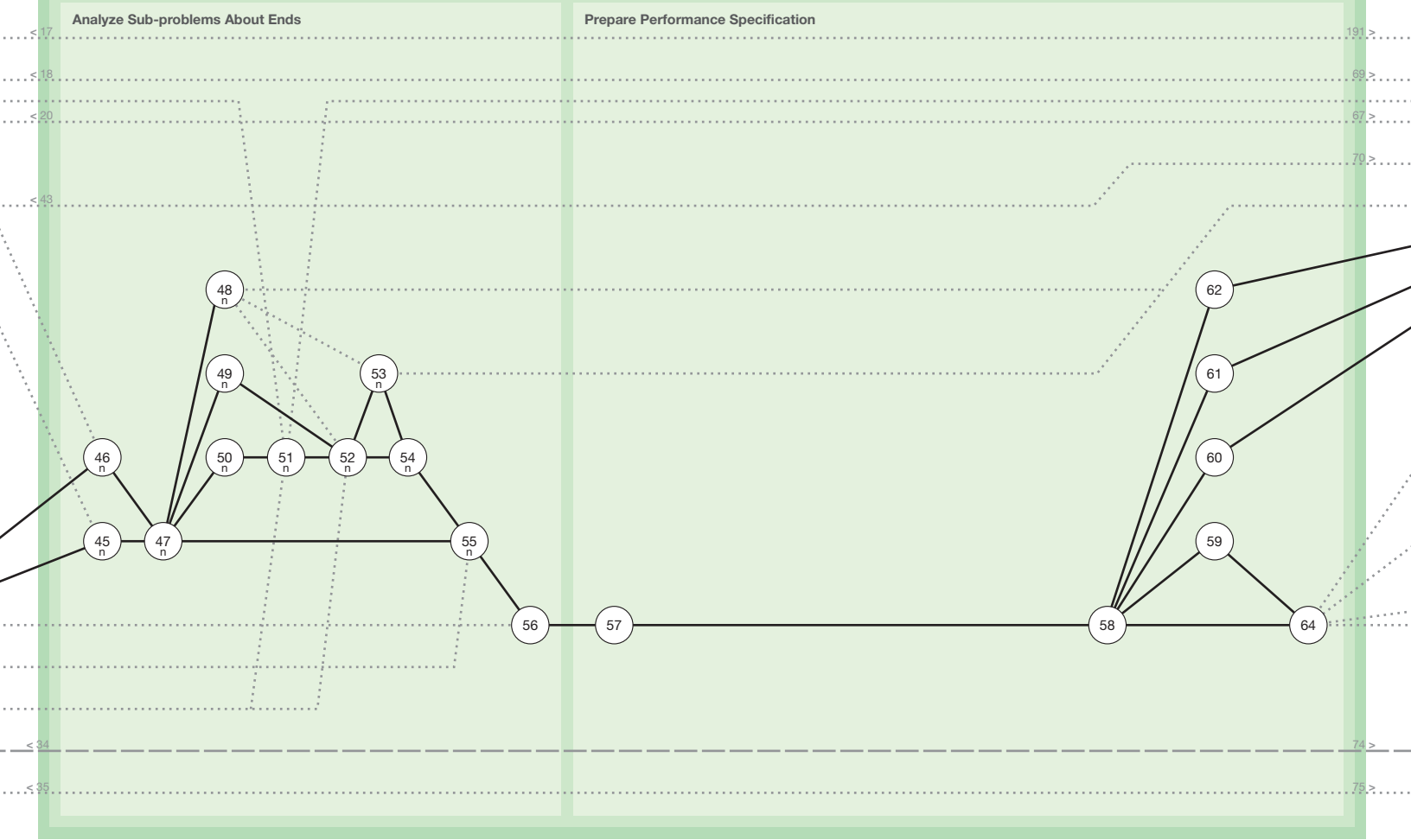
Item	Activity	Activity No	Event	
1	**Briefing**			
1.0	**Receive Instructions**	10-12	12	Commission Received To Execute Phase 1
1.0.1	Send Acknowledgment	12-13	13	Acknowledgment Dispatched
1.0.2	Bring Files And Progress Machinery Up To Date	11, 12, 13-14	14	File And Progress Machinery Up To Date
1.0.3	Make Arrangements For Formal Briefing	12-15	15	Arrangements Complete For Formal Briefing
1.1	**Define Goals (For Example):**	15-16	16	Definition Of Goals Complete
1.1.1-1.1.6	Define Client's Corporate Policy, Trading Policy, Project Aims, Problem Aims, Identify Reasons For Examining Problem Now, Define Design Goals			
1.2	**Define Constraints (For Example):**			
1.2.1-1.2.8	Identify Any National Constraints, Any Trade Constraints, Any Mandatory Company Constraints, Any Contractual Constraints, Budgetary Constraints, Marketing Constraints, Manufacturing Constraints, Any Other Constraints	15-17	17	Definition Of Constraints Complete
2	**Programming**			
2.1	**Establish Crucial Issues**			
2.1.1	Analyze Goals Recorded Under 1.1 And Define Criteria For Measuring Success	16-18	18	Criteria For Measuring Success Identified
2.1.2	Analyze Constraints Identified Under 1.2 And Define Field Available For Maneuver	17-19	19	Field For Maneuver Defined
2.1.3	Identify Crucial Issues	18, 19-20	20	Crucial Issues Identified
2.2	**Propose A Course Of Action**			
2.2.1	Review Experience Of Analogous Problems	20-21	21	Review Of Experience With Analogous Problems Complete
2.2.2	Collect Case Histories Of Similar Problems Handled Elsewhere	20-22	22	Collection Of Case Histories Ready
2.2.3	List Courses Of Action Available	21, 22-23	23	Available Courses Of Action Listed
2.2.4	Select A Promising Course Of Action	20, 23-24	24	A Promising Course Of Action Selected
2.2.5	Test Selected Course Of Action (A Pilot Study?)	24-25	25	Test Of Selected Course Of Action Completed
2.2.6	In The Light Of Experience With Analogous Problems, Appraise Probable Adequacy Of Course Of Action Selected	21, 25-26	26	Appraisal Of Probable Adequacy Complete
	If Necessary, Reiterate From 2.2.1 Until A Sufficiently Assuring Course Of Action Is Selected			
2.2.7	Reappraise Office Workload And Project Facilities	26-27	27	Reappraisal Of Office Work Load And Project Facilities Complete
2.2.8	Reappraise Program And Timetable	26-28	28	Reappraisal Of Project Program And Timetable Complete
2.2.9	Formulate Draft Revise Proposal And Estimate (With Special Reports On 1.1, 1.2 And 2.1 If Necessary)	21, 27, 28-29	29	Draft Revised Proposal And Estimate Ready



Item	Activity	Activity No	Event	
2.2.10	Check Draft Proposal And Estimate If Necessary, Reiterate From 2.2.7 Until A Satisfactory Proposal Is Drafted	29-30	30	Draft Revised Proposal And Estimate Approved
2.2.11	Prepare And Dispatch Fair Copy Of Proposal And Estimate	13, 30-31	31	Revised Proposal And Estimate Dispatched
2.2.12	Bring File And Progress Machinery Up To Date Reiterate Sections 1 And 2, Until Agreement Is Reached To Carry Out Phase 2 - "Collect Data, Prepare Performance Specification, Reappraise Proposed Program And Estimate", Or Until Project Is Abandoned.	14, 31-32	32	Files And Progress Machinery Up To Date
3	**Data Collection**			
3.0	**Receive Instructions**	31-33	33	Commission Received to Execute Phase 2
3.0.1	Send Acknowledgment	33-34	34	Acknowledgment Dispatched
3.0.2	Bring Files And Progress Machinery Up To Date	32, 33, 34-35	35	Files And Progress Machinery Up To Date
3.1	**Collect Readily Available Information (For Example On):**	33-36	36	Available Information Ready
3.1.1- 3.1.19	Environmental At Point Of Use, Identity Of Users, User Ergonomics, User Motivation, Product Function, Product Mechanics, Product Finish, Product Aesthetics, Market Environment, Competitive Products, Product Archetype, Brand Predecessors, Maker's House Style, Ruling Market Prices, Economic Quantities, Product Facilities, Limiting Dimensions, Materials, Collect Other Readily Available Information			
3.2	**Classify And Store Data**			
3.1.1	List Information Topics Handled	33-37	37	List Of Information Ready
3.1.2	Rationalize Topic Headings	36-37		
3.1.3	Design An Information Classification System	37-38		
3.1.4	Classify All Information Held			
3.1.5	Shelve The Information Gathered	36-38	38	Available Information Classified And Shelved
4	**Analysis**			
4.1	**Identify Sub-problems**			
4.1.1	List All The Factors In The Overall Problem	38-39	39	List Of Factors Ready
4.1.2	Pair Interdependent Factors So As To Form All Matters Into Sub Problems	39-40	40	Interdependence Matrix Ready
4.1.3	List All The Sub-problems Thus Identified	40-41	41	List Of Sub-problems Ready
4.1.4	Identify The Apparent Sequence Of Dependence Of Sub-problems Upon One Another	20, 41-42	42	Sub-Problem Network Ready

Phase 2 (Continued)

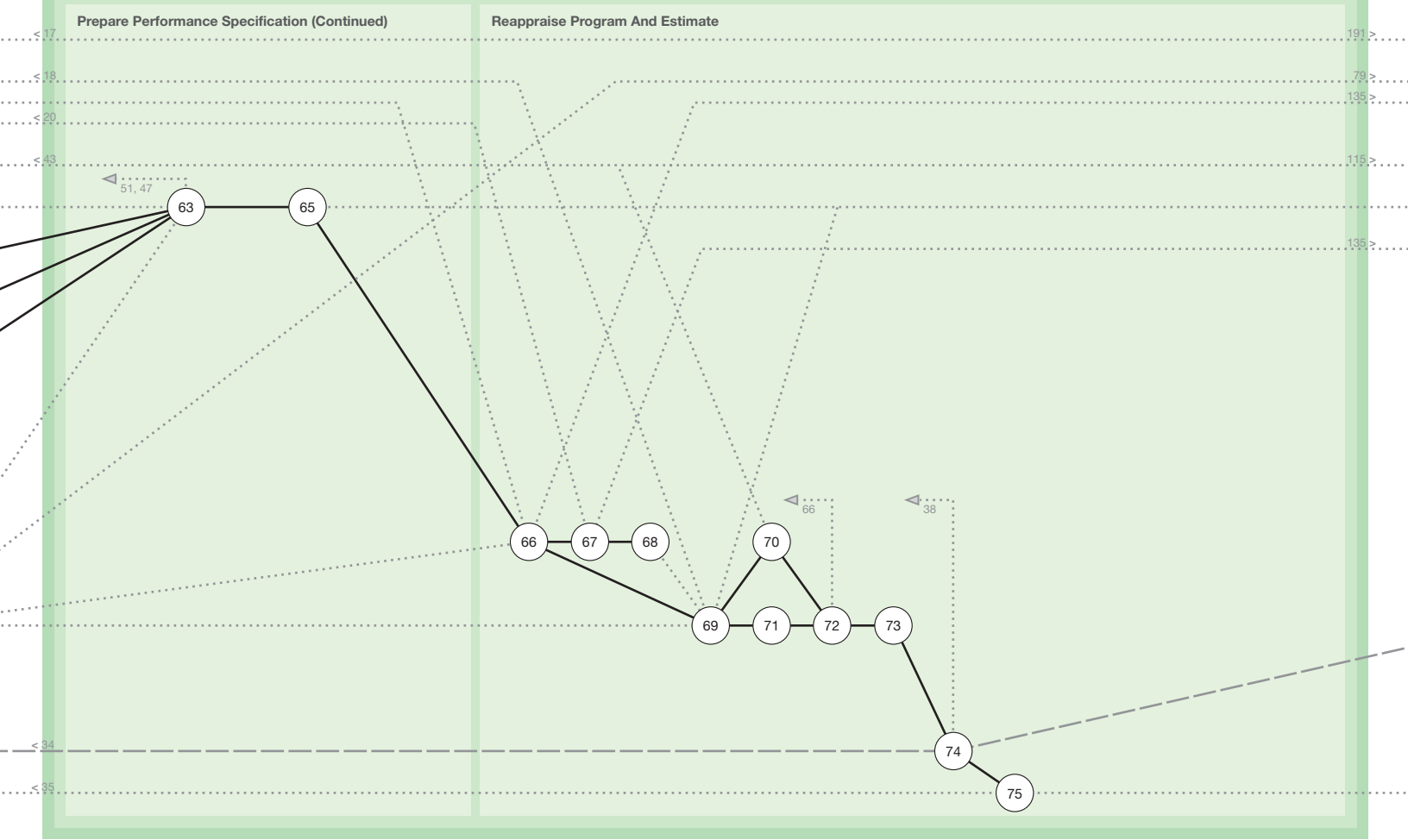
Analysis (Continued)



Item	Activity	Activity No	Event	
4.1.5	Distinguish Problems About Means From Problems About Ends	42-43, 42-44	43 44	List of Problems About Means Ready List Of Problems About Ends Ready
4.1.6	Resolve Problems About Ends As Indicated In Sections 4.2 And 4.3	44-56		
4.2	**Analyze Sub-problems About Ends**			
4.2.1	Select A Problem About Ends From The Network Identified Under 4.1.4			
4.2.2	List The Factors In This Sub-problem	17, 44-45	45	List Of Factors In Sub-problem Ready
4.2.3	Identify The Goal To Be Achieved	18, 44-46	46	Goals And Conditions Identified
4.2.4	Establish The Connection Between The Factors And The Goal	45, 46-47	47	Connection Between Factors And Goals Or Conditions Established
4.2.5	Identify Those Factors Where The Data Values May Be Voluntarily Fixed By The Designer	47-48	48	Voluntarily Evaluable Factors Identified
4.2.6	Identify Those Factors Where The Data Values Are Fixed By External Influences	47-49	49	Externally Influenced Factors Identified
4.2.7	Identify Those Factors Where The Data Are Dependent Variables (For Example, Where The Data Are The Solutions To Other Sub-problems)	47-50	50	Dependently Variable Factors Identified
4.2.8	If Necessary, Revise Network In 4.1.4 So That Problems Which Provide Data For Other Problems Are Dealt With In The Right Order, Or Select Another Problem At 4.2.1	42, 50-51	51	Revised Network Ready
4.2.9	Collect Necessary Data (Either Extract Data From Record System 3.2.5 Or Add Fresh Data)	38, 48, 49, 51-52	52	Necessary Data Ready
4.2.10	Where Sufficient, Precise Data Is Available, Delineate Maximum Field Of Feasible Solutions	52-54		
4.2.11	Where Sufficient Precise Data Is Not Available, Postulate Simplifying Assumptions By Plausible Reasoning And Then Delineate A Field Of Feasible Solutions	48, 52-53, 53-54	53 54	Simplifying Assumptions Postulated Field Of Feasible Solutions Delineated
4.2.12	Referring To 4.2.4, Identify The Zone Where Satisfaction Of Goal Is Optimum	47, 54-55	55	Optimal Solution Identified
	Reexamine Section 4.1 And Modify Problem Network As Necessary			
4.2.13	Reiterate Section 4.2 Of Each Problem About Ends In The Network	44, 55-56	56	All Problems About Ends Resolved
4.3	**Prepare Performance Specification**			
4.3.1	Taking Every Combination Of Pairs Of Sub-problems About Ends, Identify Which In Each Pair Must Take Precedence If Their Optimum Solutions Are Incompatible	56-57	57	Priorities Identified (Ranking Matrix)
4.3.2	Rank The Complete List Of Sub-problems In Order Of Precedence	57-58	58	Ranked List Of Problems About Ends Ready
4.3.3	Identify Those Pairs Where The Optimum Solutions Are In Fact Mutually Compatible, Referring To 4.2.1.2 For Each Sub-problem	58-59	59	Problems With Compatible Optimal Solutions Identified
4.3.4	Identify Those Paris Where The Optimum Solution Of One Is Compatible With Feasible (But Not The Optimum) Solutions Of The Other	58-60	60	Problems With Compatible Optimal/Feasible Solutions Identified
4.3.5	Identify Those Pairs Where Feasible Solutions (But Not The Optimum Solutions) Are Compatible	58-61	61	Problems With Compatible Feasible Solutions Identified
4.3.6	Identify Those Pairs Where Feasible Solutions And Optimum Solutions Are Incompatible	58-62	62	Problems With Incompatible Solutions Identified

Phase 2 (Continued)

Analysis (Continued)

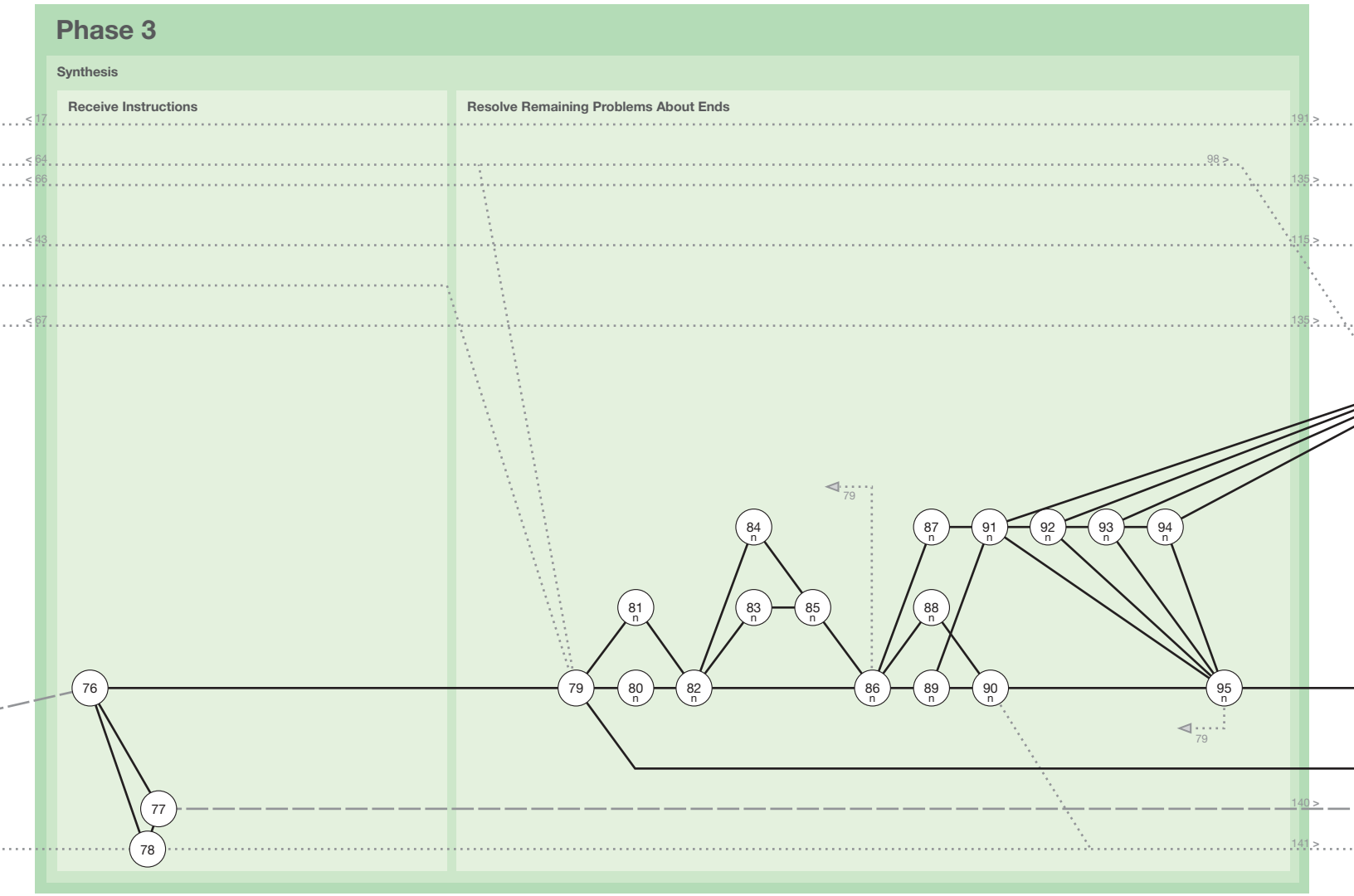


Item	Activity	Activity No	Event	
4.3.7	Select All The Sub-problems Listed Under 4.3.4 And 4.3.5 Which Stand High On The Rank Ordered List, And Reexamine Them According To Selection 4.2. Reexamine Assigned Values And Simplifying Assumptions At 4.2.11 Reiterate From 4.2.8 As Necessary. If Too Many Incompatibilities Remain, Reiterate Sections 1 And 2 Seeking Easements.	48, 53, 60, 61, 62-63	63	Critical Problems Reexamined
4.3.8	When All Problems About Ends Have Been Resolved Or As Many Of Them As Seems Practicable), Assemble All Their Solutions Into A Performance (Or Design) Specification, Ranked Substantially In Accordance With 4.3.2	58, 59, 63-64	64	Performance Specification Ready
4.3.9	List Any Remaining Intractable Problems About Ends For Reference To Client And/or For Resolution As Creative Problems	63-65	65	Remaining Intractable Problems About Ends Listed
4.4	**Reappraise Program And Estimate**			
4.4.1	In The Light Of 4.1, 4.2 And 4.3, Restate The Problem Set Out In 1.1	51, 64, 65-66	66	Problem Restatement Ready
4.4.2	Reappraise The Crucial Issues Set Out In 2.1	20, 66-67	67	Reappraised Crucial Issues Listed
4.4.3	Reappraise And If Necessary Reformulate A Course Of Action, Perviously Set Out In 2.2	67-68	68	Course Of Action Reformulated
4.4.4	Where Required, Prepare A Report On The Overall Problems About Ends, Referring To 4.3.9	18, 64, 65, 66, 68-69	69	Report On Problem About Ends Ready
4.4.5	Reappraise Program And Timetable	43, 69-70	70	Program And Timetable Reappraised
4.4.6	Reappraise Office Workload And Project Facilities	69-71	71	Workload And Facilities Reappraised
4.4.7	Formulate Draft Revised Proposal And Estimate (With Performance Specification For Approval, And Report On Overall Problem And Ends Where Necessary)	70, 71, 72	72	Draft Revised Proposal, Performance Specification And Estimate Ready
4.4.8	Check And Approve Draft Proposal And Estimate If Necessary Reiterate From 4.4.1 Until A Satisfactory Proposal Is Drafted	72-73	73	Draft Proposal And Estimate Dispatched
4.4.9	Prepare And Dispatch Fair Copy Of Proposal And Estimate	34, 73-74	74	Proposal And Estimate Dispatched
4.4.10	Bring Files And Progress Machinery Up To Date	35, 74-75	75	Files And Progress Machinery Up To Date

Reiterate Section 4 Until Agreement Is Reached To Carry Out Phase 3 - "Prepare Outline Design Proposal(s)", Or Project Is Terminated.

Where Suitable, Phase 4 - "Develop Prototype Design(s)" And/or Phase 5 - "Prepare (And Execute) Validation Studies" May Also Be Commissioned At The Same Time As Phase 3.

Phase 3



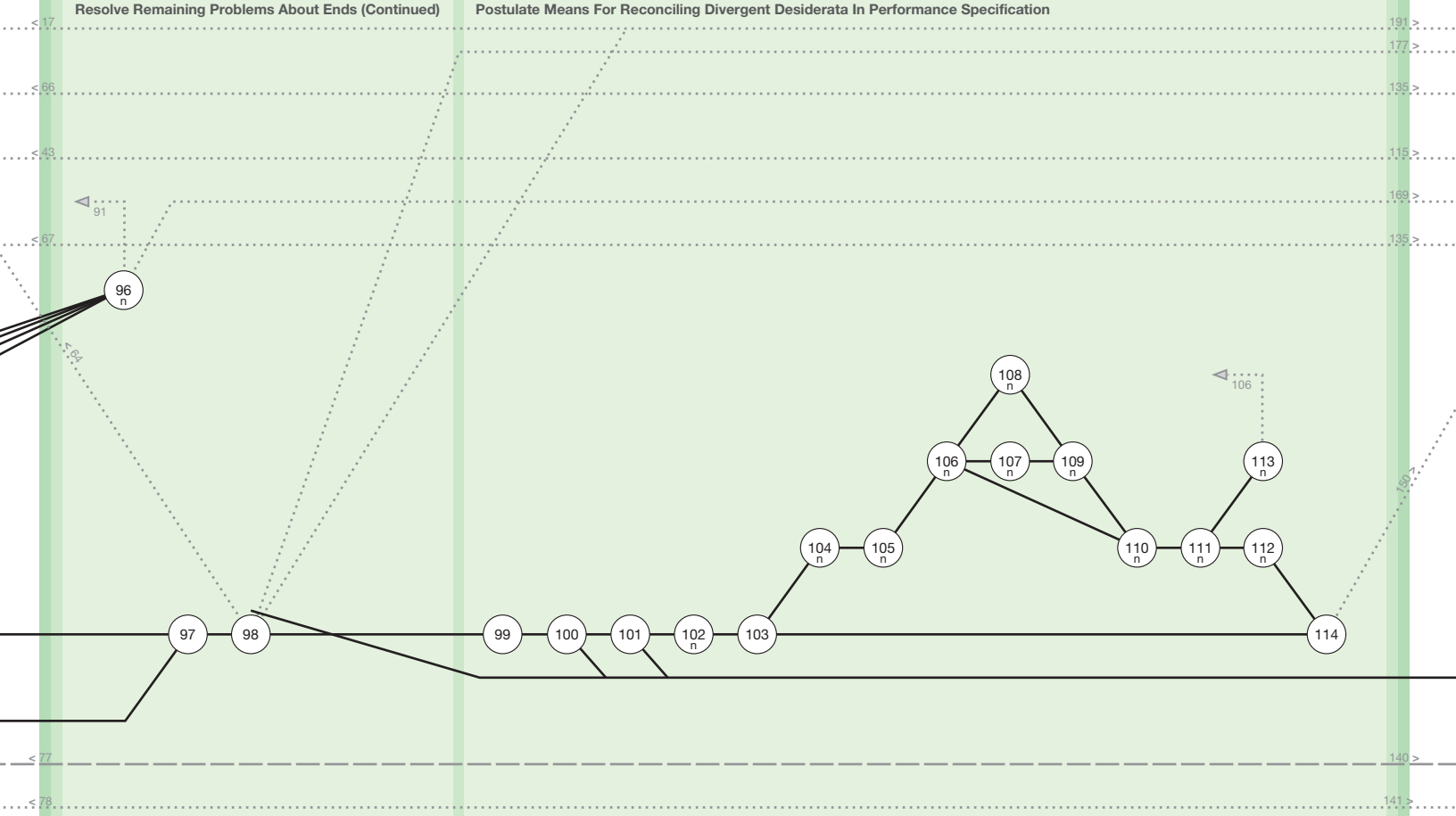
Item	Activity	Activity No	Event	
5	**Synthesis**			
5.0	**Receive Instructions**	74-76	76	Commission Received To Execute Phase 3
5.0.1	Send Acknowledgment	76-77	77	Acknowledgment Dispatched
5.0.2	Bring Files And Progress Machinery Up To Date	75, 76, 77-78	78	Files And Progress Machinery Up To Date
5.1	**Resolve Remaining Problems And Ends (Note That, In General, Solutions To Problems About Ends Will Pose Problems About Means)**			
5.1.1	Reappraise The Performance Specification Prepared Under 4.3.8 And The List Of Intractable Problems About Ends Prepared Under 4.3.9 And Prepare A New List Of Unresolved Problems About Ends	64, 65, 69, 76-79	79	Revised List Of Unresolved Problems About Ends Ready
5.1.2	For Each Problem In The List Prepare Under 5.1.1, List The Factors In The Problem	79-80	80	List Of Factors In Sub-problem Ready
5.1.3	Identify The Goals To Be Achieved And The Constraints Or Conditions To Be Satisfied	79-81	81	Goals And Constraints Or Conditions For Sub-problems Identified
5.1.4	Establish The Connections Between The Factors (Or The Goals And Constraints Or Conditions)	80, 81-82	82	Connections Between Factors Established
5.1.5	Identify Similar Or Analogous Problems In Prior Experience	82-83	83	Analogous Problems In Prior Experience Identified
5.1.6	Identify Similar Or Analogous Problems Handled Elsewhere	82-84	84	Analogous Problems Handled Elsewhere Identified
5.1.7	Catalog The Properties Of The Analogous Problems And Reexpress Each Within A Common Format	83, 84-85	85	Analysis Of Analogous Problems Complete
5.1.8	Reexpress Present Sub-problem Within The Format Developed Under 5.1.7	82, 85-86	86	Re-expression Of Sub-problem Within Common Format Complete
5.1.9	Identify Those Factors In The Sub-problem For Which The Data Values May Be Voluntarily Fixed By The Designer	86-87	87	Factors Where Data Values Are Voluntarily Assignable Identified
5.1.10	Identify Those Factors In The Sub-problem For Which The Data Values Are Fixed By External Influences	86-88	88	Factors Where Data Values Are Externally Fixed Identified
5.1.11	Identify Those Factors Where The Data Are Dependent Variables (For Example, Where The Data Are The Solutions To Other Sub-problems). If Necessary, Suspend Work On This Problem And Select Another At 5.1.1 So That Sub-problems Are Dealt With In The Right Order	86-89	89	Factors Where Data Are Dependent Variables Identified
5.1.12	Collect Necessary Data (Either Extract Data From Record System 3.2.5 Or Add Fresh Data)	88, 89-90	90	Necessary Data Ready
5.1.13	Where Sufficient Precise Data Is Not Available, Postulate Simplifying Assumptions Or Assign Values By Plausible Reasoning	87, 88, 89-91	91	Simplifying Assumptions Postulated And/or Data Values Assigned
5.1.14	Where Not Practical Solution Emerges, Vary One Of The Voluntarily Assigned Values And/or Assumptions (See 5.1.9 And 5.1.13) And Seek Easements	90, 91-92	92	No Solution; Assigned Values And/or Simplifying Assumptions Varied
5.1.15	Where This Fails, Reappraise Constraints (See 5.1.10), And Seek Easements	92-93	93	No Solution; Constraints Or Conditions Eased
5.1.16	In The Last Resort, Reappraise Goals (See 5.1.3) And Seek Variation	93-94	94	No Solution; Goals Varied
5.1.17	Resolve Problem, Delineating Maximum Field Of Feasible Solutions	90, 91, 92, 93, 94-95	95	Solution To Sub-problem Ready

Phase 3 (Continued)

Synthesis (Continued)

Resolve Remaining Problems About Ends (Continued)

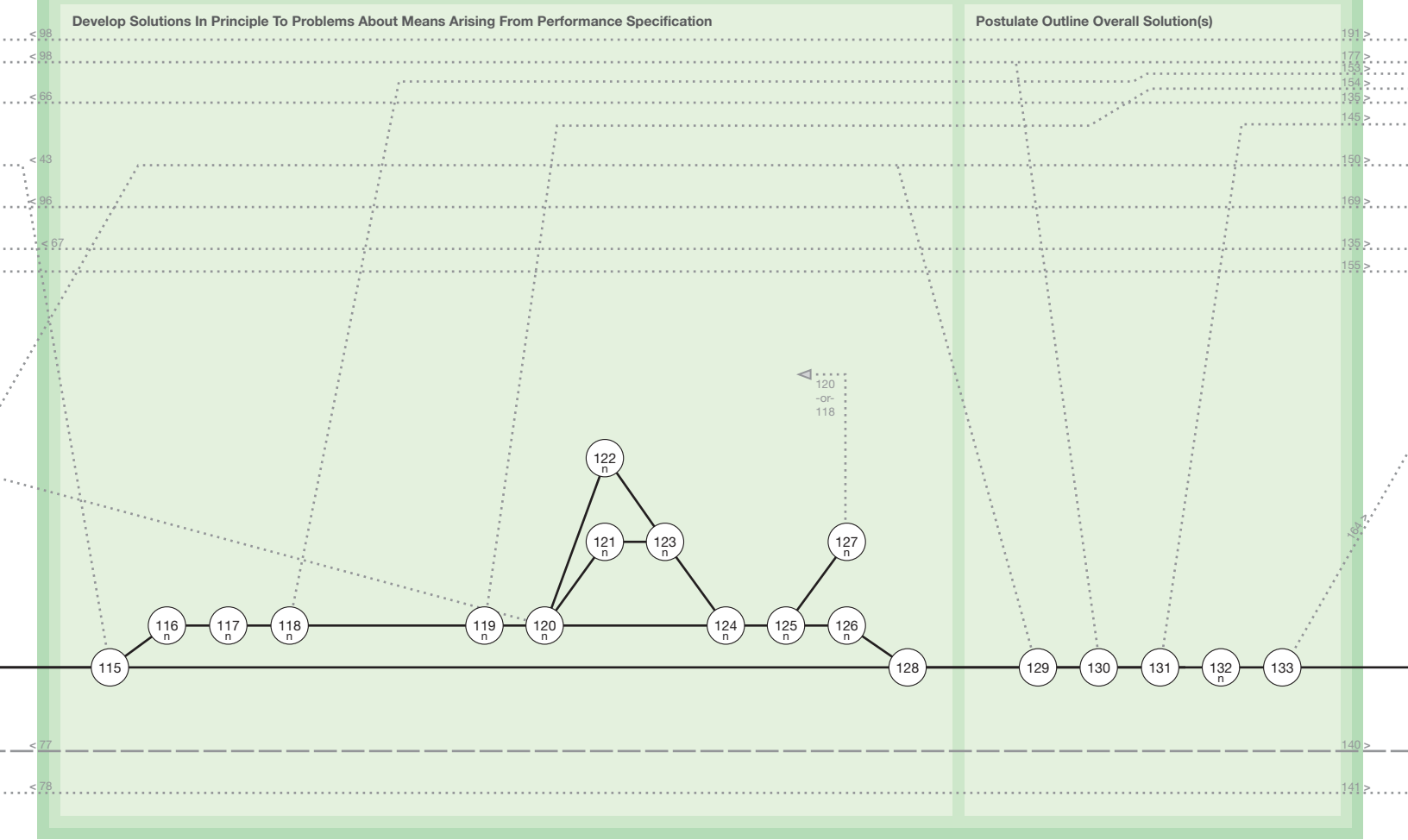
Postulate Means For Reconciling Divergent Desiderata In Performance Specification



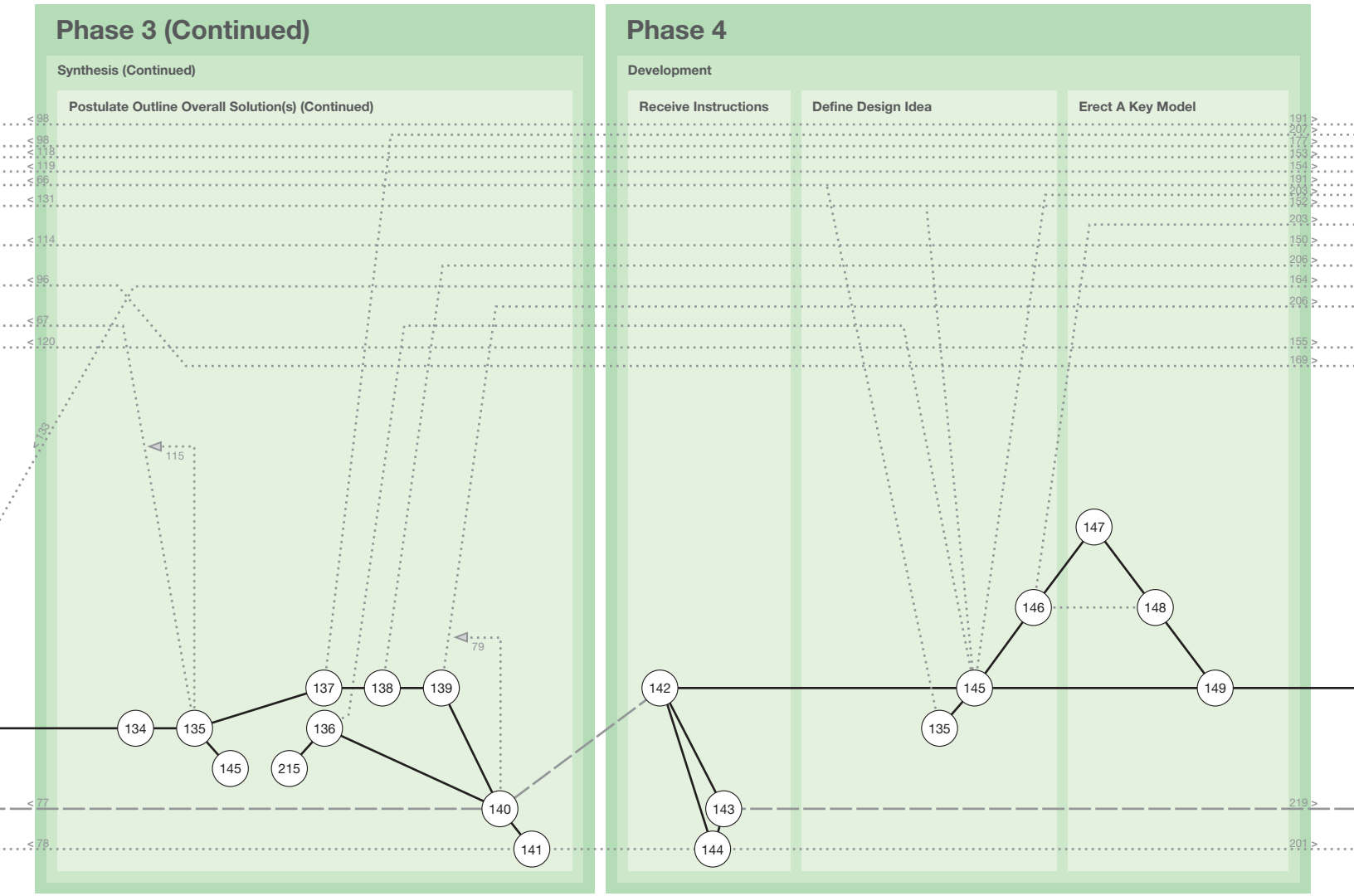
Item	Activity	Activity No	Event	
5.1.18	If A Variation Or Assumption Has Been Made Under 5.1.13-16 And A Solution Has Emerged, Refer To Source And Check Prima Facie Acceptability Of Variation Or Assumption. (See Also Validation Studies, Section 6.5.) Where Necessary, Reiterate From 5.1.14	91, 92, 93, 94, 95-96	96	Prima Facie Validation Of Assumptions And Variations Complete
5.1.19	Reiterate From 5.1.2 Until All Outstanding Problems About Ends Listed Under 5.1.1 Are Resolved	79, 95-97	97	All Problems About Ends Resolved
5.1.20	Unite Solutions Prepared Under 5.1.19 With Performance Specification Prepared Under 4.3.8 To Form A Revised Specification.	64, 97-98	98	Revised Performance Specification Ready
5.2	**Postulate Means For Reconciling Divergent Desiderata In Performance Specification**			
5.2.1	Examine The Augmented Performance Specification Prepared Under 5.1.20, And Identify And Group Those Desiderata Which Appear To Be Inter-related	98-99	99	Inter-related Desiderata Identified (Interaction Matrix)
5.2.2	List The Groups Of Inter-related Desiderata	99-100	100	List Of Inter-related Desiderata Ready
5.2.3	From The List Prepared Under 5.2.2, Select Those Groups Containing Desiderata Which Appear To Be Divergent Or Contradictory	100-101	101	Groups Containing Divergent Desiderata Identified
5.2.4	For Each Group Selected Under 5.2.3, Reexpress The Desiderata As The Goals And Constraints Or Conditions For One Or More Problems About Means	101-102	102	Divergent Desiderata Reexpressed As Goals
5.2.5	List The Problems About Means Thus Identified	102-103	103	List Of Problems About Means Ready
5.2.6	For Each Problem About Means Listed Under 5.2.5, List The Factors In The Problem	103-104	104	List Of Factors In The Sub-problem Ready
5.2.7	Identify Goals And Constraints Or Conditions In The Sub-problem	104-105	105	List Of Goals And Constraints Or Conditions Read
5.2.8	Establish The Connections Between The Factors (Goals And Constraints Or Conditions)	105-106	106	Connection Between Factors And The Goals And Constraints Or Conditions Established
5.2.9	Identify Similar Or Analogous Problems In Prior Experience	106-107	107	Analogous Problems In Prior Experience Identified
5.2.10	Identify Similar Or Analogous Problems Handled Elsewhere	106-108	108	Analogous Problems Handled Elsewhere Identified
5.2.11	Catalog The Properties Of The Nearest Analogous Problems And The Elements Of Their	107, 108-109	109	Problem Element/solution Matrix Ready
5.2.12	Respective Solutions And Prepare A Problem Element/problem Solution Matrix Reexpress The Present Problem Within The Same Matrix	106, 109-110	110	Re-expression Of Sub-problem With Same Matrix Complete
5.2.13	Explore Combinations Of Solution Elements	110-111	111	Exploration Of Combinations Of Solution Elements Complete
5.2.14	Select A Promising Combination Of Solution Elements As A Hypothesis For Development	111-112	112	Hypothesis Selected
5.2.15	Where No Promising Hypothesis Emerges, Reiterate From 5.2.6 In Wider Fields Reiterate From 5.2.6 Until All Problems About Means Arising From Divergent Desiderata In The Performance Specification Are Resolved	111-113	113	No Hypothesis
5.2.16	Assemble Hypothetical Solutions	103, 112-114	114	Solutions-in-principle Assembled

Phase 3 (Continued)

Synthesis (Continued)



Item	Activity	Activity No	Event
5.3	**Develop Solutions In Principle To Problems About Means Arising From Performance Specification**		
5.3.1	Referring To The Performance Specification Prepared Under 5.1.20, The List Of Groups Of Interrelated Desiderata Prepared Under 5.2.3, List Those Groups And Single Desiderata Not Yet Handled. Add Any Problem About Means Remaining From Those Under 4.1.5	43, 98, 100, 101-115	115 List Of Outstanding Inter-related Desiderata Ready
5.3.2	For Each Item Listed Under 5.3.1, Reexpress The Goals And Constraints For One Or More Problems About Means	115-116	116 Goals And Constraints For Problems About Means Identified
5.3.3	List The Problems About Means Thus Identified	116-117	117 List Of Problems About Means Ready
5.3.4	For Each Problem About Means Listed Under 5.3.3, List The Factors In The Problem	117-118	118 List Of Factors In The Sub-problem Ready
5.3.5	Identify Goals And Constraints Or Conditions In The Sub-problem	118-119	119 Revised List Of Goals And Constraints Or Conditions Ready
5.3.6	Establish The Connections Between The Factors (Goals And Constraints Or Conditions)	119-120	120 Connection Between Factors And The Goals And Constraints Or Conditions Established
5.3.7	Identify Similar Or Analogous Problems In Prior Experience	120-121	121 Analogous Problems In Prior Experience Identified
5.3.8	Identify Similar Or Analogous Problems Handled Elsewhere	120-122	122 Analogous Problems Handled Elsewhere Identified
5.3.9	Catalog The Properties Of The Nearest Analogous Problems And The Elements Of Their Respective Solutions And Prepare A Problem Element/problem Solution Matrix	121,121-123	123 Problem Element/solution Matrix Ready
5.3.10	Reexpress The Present Problem Within The Same Matrix	120, 123-124	124 Re-expression Of Sub-problem With Same Matrix Complete
5.3.11	Explore Combinations Of Solution Elements	124-125	125 Exploration Of Combinations Of Solution Elements Complete
5.3.12	Select A Promising Combination Of Solution Elements As A Hypothesis For Development	125-126	126 Hypothesis Selected
5.3.13	Where Not Promising Hypothesis For Development Emerges, Reiterate From 5.3.7 In Wider Field Reiterate From 5.3.4 Until All Problems About Means Arising From Desiderata In Performance Specification Are Resolved In Principle	125-127	127 No Hypothesis
5.3.14	Assemble Hypothetical Solutions	115, 126-128	128 Solutions-in-principle Assembled
5.4	**Postulate Outline Overall Solution(s)**		
5.4.1	Unite The Collection Of Hypothetical Solutions Under 5.2.16 With Those Under 5.3.14	114, 128-129	129 Combined Collection Of Hypothetical Solutions Ready
5.4.2	Using The Performance Specification Prepared Under 5.1.20 As A Guide, Take Every Combination Of Pairs Of Hypothetical Solutions And Identify Which In Each Pair Should Take Precedence If They Should Later Prove To Be Incompatible	98, 129-130	130 Ranking Matrix For Problems About Ends Ready
5.4.3	Rank The Collection Of Hypothetical Solutions In Order Of Precedence	130-131	131 Ranked List Of Hypothetical Solutions Ready
5.4.4	Taking Every Combination Of Paris Of Hypothetical Solutions From The List Under 5.4.3, Beginning With The Pairs Containing The Highest Ranked Solutions, And In The Light Of Whatever Information Is Readily Available, Check The Plausibility Of Each Pair's Proving To Be Compatible Where Probable Incompatibilities Are Identified, Reiterate From 5.3.2	131-132	132 Compatibility Studies Complete

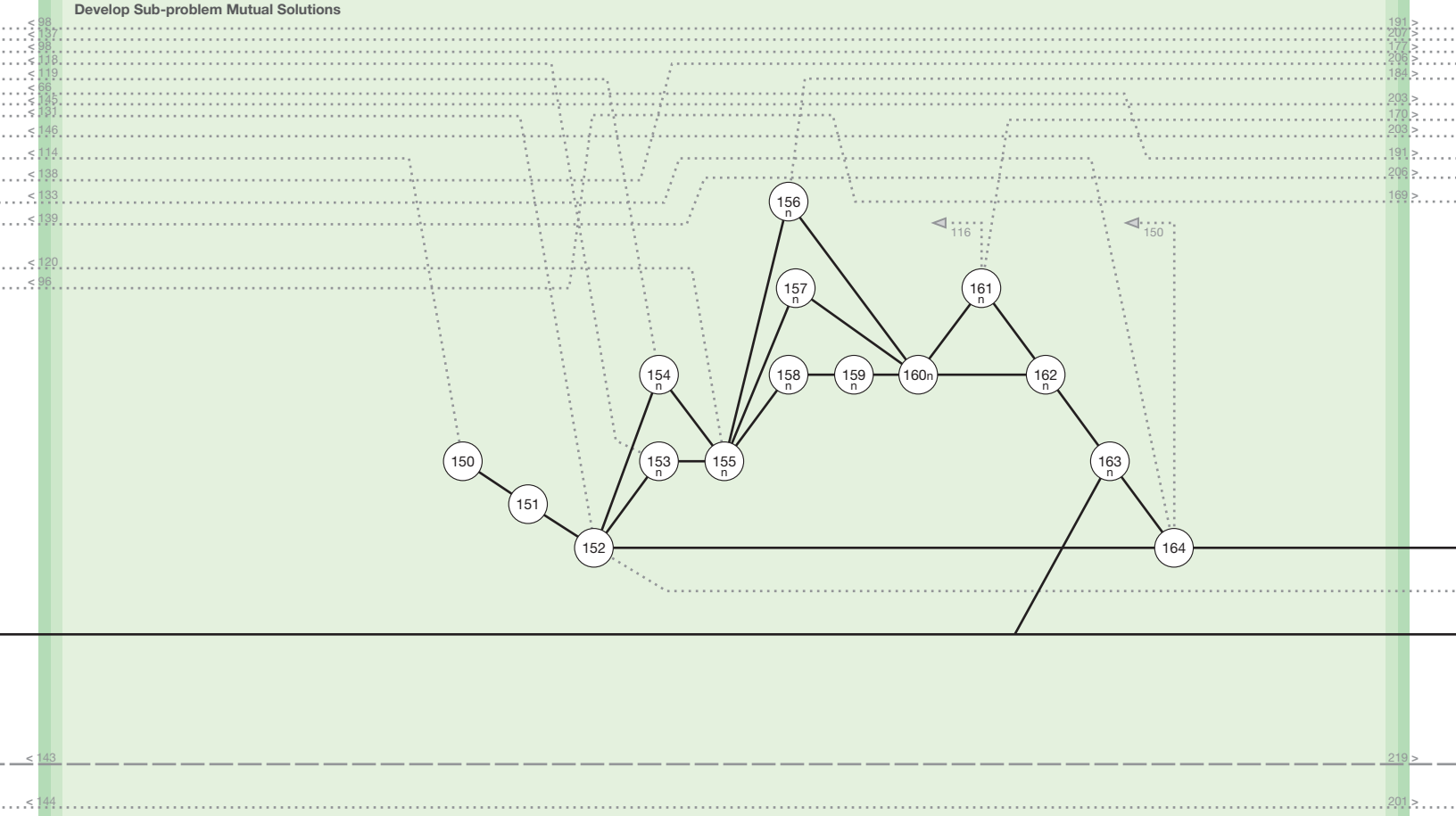


Item	Activity	Activity No	Event
5.4.5	Assemble All The Hypothetical Solutions Into A Suggested Composite Overall Solution, Or Range Of Solutions	132-133	133 Composite Overall Solution(s) Ready
5.4.6	Reappraise The Proposed Solution(s) In The Light Of The Problem As Set Out In 4.4.1	66, 133-134	134 Reappraisal In Light Of Original Problem
5.4.7	Reappraise The Proposed Solution(s) In The Light Of Crucial Issues Set Out In 4.4.2	67, 134-135	135 Reappraisal In Light Of Crucial Issues Complete
5.4.8	Where Required, Prepare And Submit Sketch Designs (See Note *Below*)	215, 135-136	136 Sketch Designs Ready
5.4.9	Reappraise And If Necessary Reformulate A Course Of Action, Previously Set Out In 4.4.3	135-137	137 Reappraisal And/or Reformulation Of Course Of Action Complete
5.4.10	Reappraise Timetable	137-138	138 Reappraisal Of Timetable Complete
5.4.11	Reappraise Facilities	138-139	139 Reappraisal Of Facilities Complete
5.4.12	If Necessary, Prepare Revised Proposals	77, 136, 139-140	140 Revised Proposals Ready
5.4.13	Bring Files And Progress Machinery Up To Date	78, 140-141	141 Files And Progress Machinery Up To Date
	Note If Sketch Designs Must Be Submitted For Approval At This Stage See 5.4.8, Continue With Section 6.1 And Then The Whole Of Section 7 For The Sketch Design Only. Reiterate The Whole Of Section 5 Until A Sketch Design Is Approved And A Commission Is Received To Execute Phase 4. Proceed With Section 6 In Respect Of The Finished Design.		
	Development		
6.0	**Receive Instructions**	140-142	142 Commission Received To Execute Phase 4
6.0.1	Send Acknowledgment	142-143	143 Acknowledgment Dispatched
6.0.2	Bring Files And Progress Machinery Up To Date	141, 142, 143-144	144 Files And Progress Machinery Up To Date
6.1	**Define Design Idea**		
6.1.1	Using The Most Abstract Or General Medium Available (eg., A Form Of Words), Completely Define The Essential Germ Of The Design Idea	131, 133, 135, 142-145	145 Definition Of Essential Germ Of Design Idea Ready
6.1.2	Using The Same Medium, Define The Maximum Variation Embraceable By The Design Idea In The Case Of Sketch Designs Being Prepared Under 5.4.8, Proceed Now With Section 7 In Respect Of Sketch Designs Only.	145-146	146 Definition Of Range Of Variation Embraceable By Design Idea Ready
6.2	**Erect A Key Model**		
6.2.1	List Range Of Media For The Representation Of An Embodiment Of The Design Idea	146-147	147 List Of Available Media Ready
6.2.2	Select The Least Abstract Medium Which Will Exhaustively Describe The Variants Of The Design Idea (See 6.1.2)	146, 147-148	148 Medium Selected
6.2.3	Erect A Key Model Of The Essential Design Idea Showing All It's Variants	145, 148-149	149 Key Model Erected

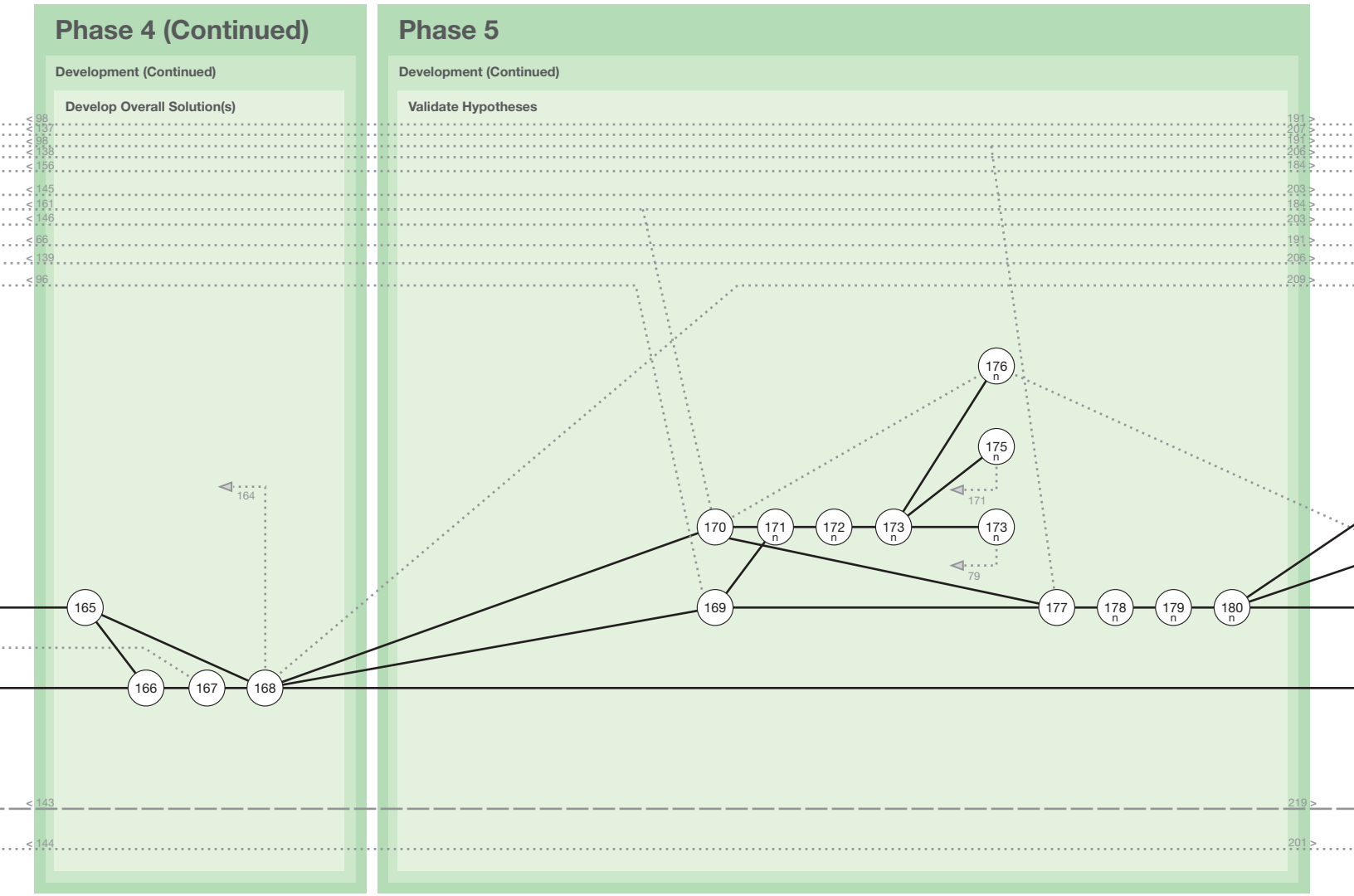
Phase 4 (Continued)

Development (Continued)

Develop Sub-problem Mutual Solutions



Item	Activity	Activity No	Event
6.3	**Develop Sub-problem Mutual Solutions**		
6.3.1	Take The Combined Collection Of Hypothetical Sub-problem Solutions Under 5.4.1 And, In The Light Of The Information Employed In Their Respective Resolutions (See Section 5.2 And Section 5.3), Identify All Combinations Of Pairs Of Hypothetical Solutions Which Appear To Involve Inter-acting Development Details.	129-150	150 Interaction Matrix Ready
6.3.2	List The Inter-acting Pairs Thus Identified	150-151	151 List Of Inter-acting Pairs Of Hypothetical
6.3.3	Using The Ranked List Of Hypothetical Sub-problem Solutions Prepared Under 5.4.3 As A Guide, Identify And Plot In The Form Of A Network The Most Desirable Sequence For Detailed Development Of The Chain Of Hypothetical Sub-problem Solutions	131, 151-152	152 Sub-problem Solutions Ready
6.3.4	Select The Earliest Undeveloped Hypothesis In The Network, Referring Where Necessary To Working Papers Prepared Under Section 5.3, And List The Factors In Developing This Hypothesis Into A Material Embodiment	118, 152-153	153 List Of Factors In Sub-problem Ready
6.3.5	Referring Where Necessary To Working Papers Under Section 5.3, Identify The Goals And Constraints Or Conditions In The Sub-problem	119, 152-154	154 List Of Goals And Constraints Or Conditions Ready
6.3.6	Referring Where Necessary To Working Papers Prepared Under Section 5.3, Establish The Connections Between The Factors (Goals And Constraints Or Conditions)	120, 153, 154-155	155 Connections Between Factors Established
6.3.7	Identify Those Factors Where The Data Values May Be Voluntarily Assigned By The Designer	155-156	156 Factors Where Data Values Are Voluntarily Assignable Identified
6.3.8	Identify Those Factors Where The Data Values Are Fixed By External Influences	155-157	157 Factors Where Data Values Are Fixed By External Influences Identified
6.3.9	Identify Those Factors Where The Data Are Dependent Variables (For Example, Where The Data Are He Solutions To Other Sub-problems)	155-158	158 Factors Where Data Values Are Dependent Variables Identified
6.3.10	If Necessary, Revise Network Prepared Under 6.3.3 So That Problems Which Provide Data For Other Sub-problems Are Dealt With In The Right Order, Or Select Another Problem At 6.3.4	158-159	159 Revised Network Ready
6.3.11	Collect Necessary Data	156, 157, 159-160	160 Necessary Data Ready
6.3.12	Where Sufficient, And Sufficiently Precise, Data Is Available, Develop A Model Embodiment Of The Hypothetical Solution	156, 160-162	
6.3.13	Where Sufficient Precise Data Is Not Available, Postulate Simplifying Assumptions By Plausible Reasoning And Develop A Model Embodiment Of The Hypothetical Solution	156, 160-161	161 Simplifying Assumptions Postulated 162 Material Embodiment Ready
6.3.14	Test The Embodiment For Fit Within The Framework Of The Key Model Where Embodiment Is Found To Be Impracticable, Revise Hypothetical Solution By Reiterating Section 5.3 For That Sub-problem, And Section 5.4 For The Effect Of A New Hypothesis On The Overall Design Concept	149, 162-163	163 Test Of Sub-problem Solution Embodiment With Key Model Complete
6.3.15	Reexamine The Inter-action Matrix Prepared Under 6.3.1 And The Network Prepared Under 6.3.3, Revise Network As Necessary And Select Another Sub-problem	133, 152, 163-164	164 All Hypothetical Sub-problem Solutions Embodied

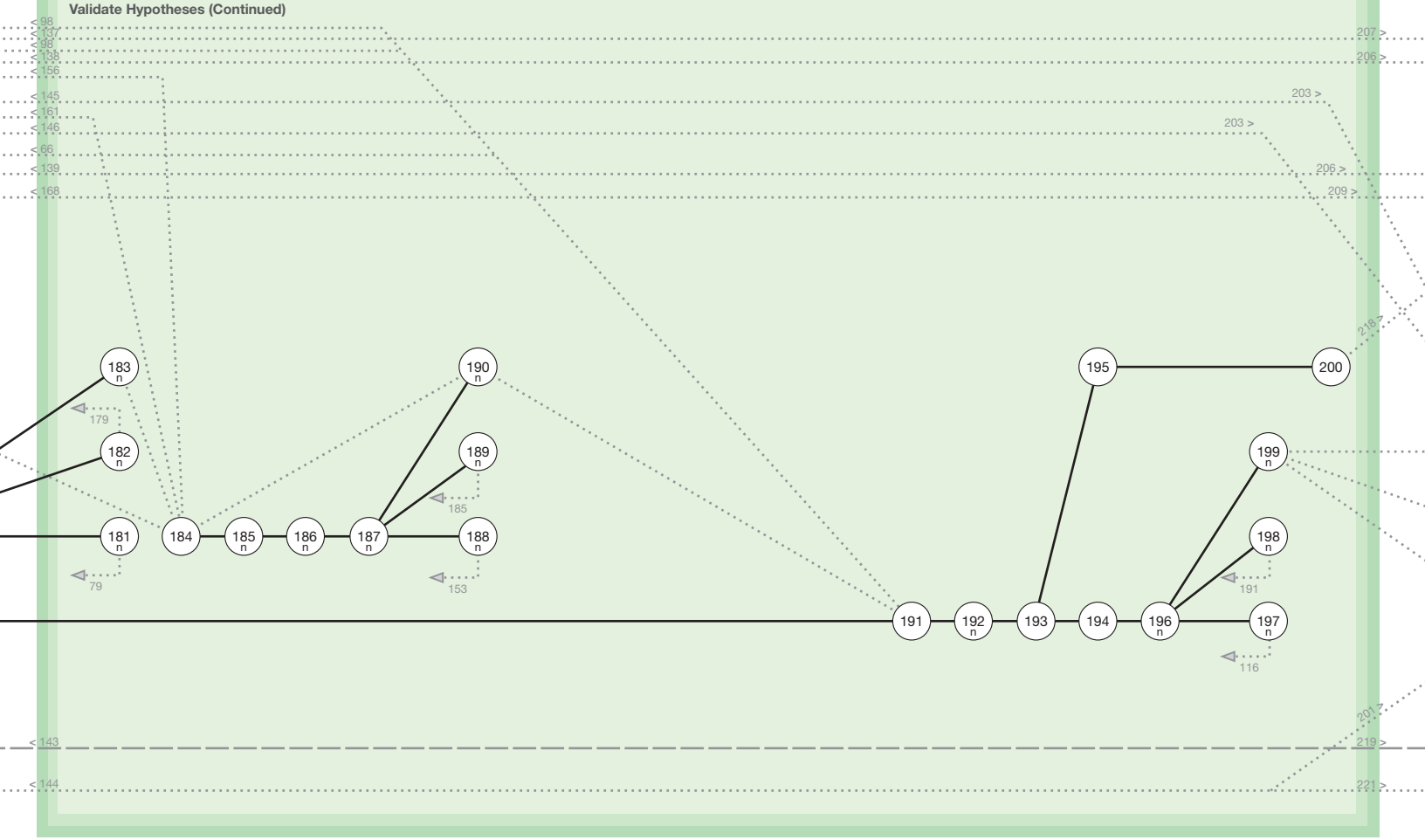


Item	Activity	Activity No	Event
6.4	**Develop Overall Solutions**		
6.4.1	Unite Model Sub-problem Embodiments Into One Or More Model Overall Solution(s)	164-165	165 Model Overall Solution Ready
6.4.2	Identify Undeveloped Areas In The Design	149, 165-166	166 Underdeveloped Area Identified
6.4.3	Reexpress As A Sequence Of Additional Problems For Completion	152, 166-167	167 New Networks Ready
6.4.4	Develop Each Problem Under 6.4.3 By Reiterating From 6.3.4 Reiterate Section 6.4 Until One Or More Overall Solutions Are Fully Developed	165, 167-168	168 Overall Solution Ready
6.4.4	Develop Each Problem Under 6.4.3 By Reiterating From 6.3.4 Reiterate Section 6.4 Until One Or More Overall Solutions Are Fully Developed	165, 167-168	168 Overall Solution Ready
6.5	**Validate Hypotheses**		
6.5.1	Select And List Those Problems About Ends Where Data Values Were Voluntarily Assigned (See 4.2.5, 4.3.7 And 5.2.9)	96, 168-169	169 List Of Problems About Ends Containing Voluntarily Assigned Values Ready
6.5.2	Select And List Those Problems About Ends Where Simplifying Assumptions Were Made (See 4.2.11, 4.3.7 And 5.1.13)	161, 168-170	170 List Of Problems About Ends Containing Simplifying Assumptions Ready
6.5.3	For Each Problem Listed Under 6.5.1 Or 6.5.2, Formulate Hypotheses Calculated To Facilitate The Validation Of The Solution To The Problem (See 4.2.11, 4.2.12, 5.1.17 And 5.1.18)	169, 170-171	171 Hypotheses Ready
6.5.4	For Each Hypothesis Under 6.5.3, Design An Experiment To Test The Hypothesis And Validate The Solution (See Note Below)	171-172	172 Design For Experiment Or Test Ready
6.5.5	Conduct Validation Experiment Or Study As Planned Under 6.5.4	172-173	173 Experiment Or Study Complete
6.5.6	Where The Experiment Or Study Shows That The Solution To The Problem About Ends Is Invalid (See 6.5.3), Then Rework The Problem About Ends And All It's Ramifications From 5.1.2	173-174	174 Solution To Problem About Ends Shown To Be Invalid
6.5.7	Where The Experiment Or Study Is Inconclusive, Reiterate From 6.5.4	173-175	175 Result Of Experiment Or Study Inconclusive
6.5.8	Where The Experiment Or Study Indicates That The Solution To The Problem About Ends Is Supported, Reiterate From 6.5.3 Until Problems Listed Under 6.5.1 And 6.5.2 Are Validated	170, 173-176	176 All Solutions To Problems About Ends Containing Assumptions Now Validated
6.5.9	Identify And List Those Statements In The Performance Specification (See 5.1.20) Which Have Not Been Examined Under 6.5.3 Note: Where Necessary, The Problem Of Designing An Experiment Can Be Handled By The Procedures Set Out In Section 4 And Section 5. These Procedures Include Appraising The Significance Of The Problem Under Investigation And The Amount Of Effort Which Can Be Devoted To It's Solution	98, 169, 170-177	177 List Of Untested Statements In Performance Specification Ready
6.5.10	For Each Statement Listed Under 6.5.9, Formulate Hypotheses Calculated To Facilitate Validation Of The Statement	177-178	178 Hypothesis Formulated
6.5.11	For Each Hypothesis Under 6.5.10, Design An Experiment To Test The Hypothesis And Validate The Statement (See Note On Page 26)	178-179	179 Design For Experiment Or Study Ready
6.5.12	Conduct Validation Experiment Or Study As Planned Under 6.5.11	179-180	180 Experiment Or Study Complete

Phase 5 (Continued)

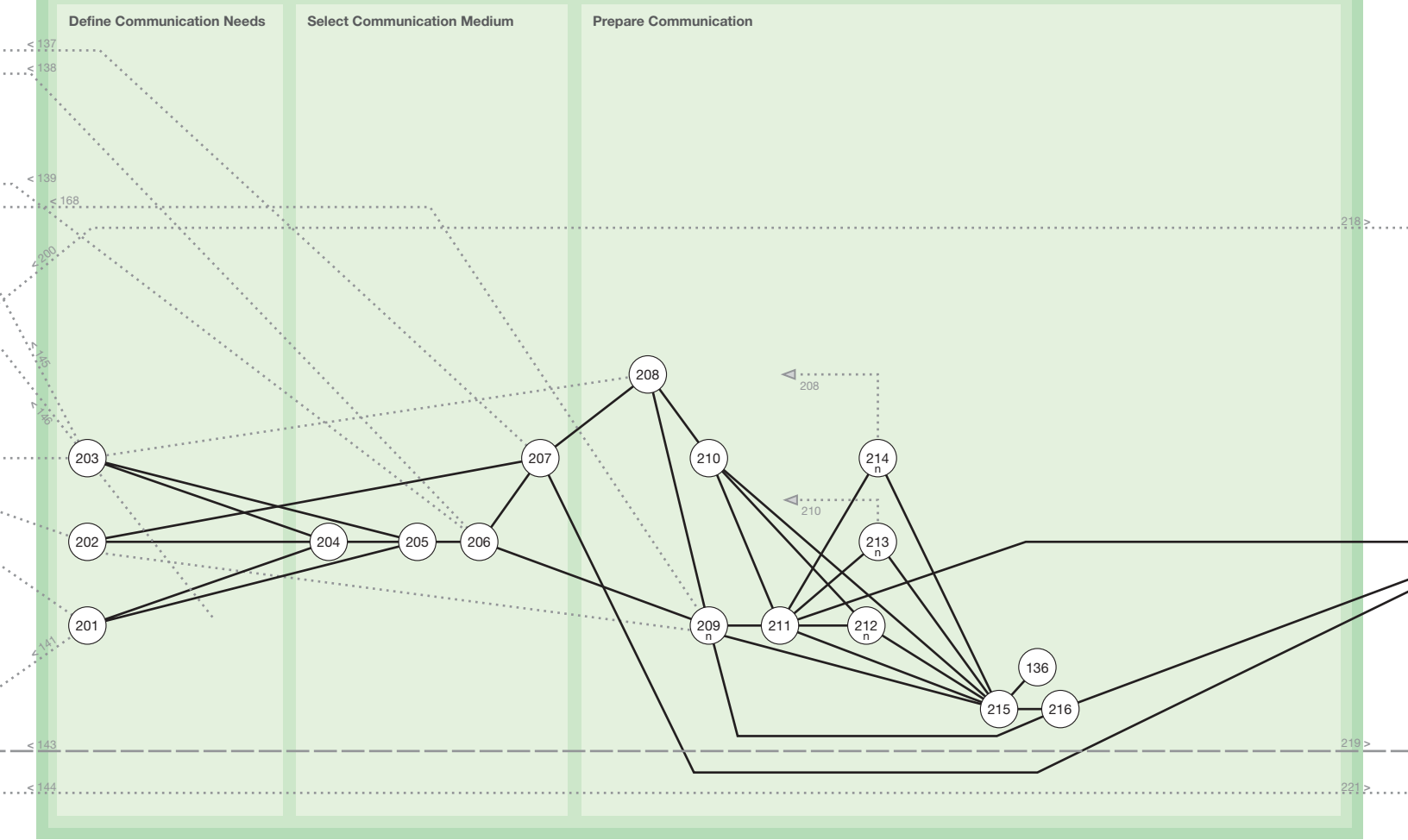
Development (Continued)

Validate Hypotheses (Continued)



Item	Activity	Activity No	Event
6.5.13	Where The Experiment Or Study Shows That The Statement In The Specification Is Invalid, Then Rework The Problems About Ends Embodied In The Statement, And All Their Ramifications, From 5.1.1	180-181	181 Specification Statement Shown To Be Invalid
6.5.14	Where The Experiment Or Study Is Inconclusive, Reiterate From 6.5.11	180-182	182 Experiment Or Study Inconclusive
6.5.15	Where The Experiment Or Study Indicates That The Specification Statement Is Supported, Reiterate From 6.5.10 Until All Statements In The Performance Specification Are Validated	177, 180-183	183 All Specification Statements Validated
6.5.16	Identify And List Those Design Details Where Data Values Were Voluntarily Assigned By The Designer (See 6.3.7), And Where Simplifying Assumptions Were Made (See 6.3.13)	156, 161, 176, 183-184	184 List Of Design Details Based On Assumptions Ready
6.5.17	For Each Detail Under 6.5.16, Formulate Hypotheses To Facilitate Validation Of The Detail	184-185	185 Hypotheses Formulated
6.5.18	For Each Hypothesis Under 6.5.17, Design An Experiment To Test The Hypothesis And Validate The Detail (See Note After 6.5.9)	185-186	186 Design For Experiment Or Study Ready
6.5.19	Conduct Validation Experiment Or Study As Planned Under 6.5.18	186-187	187 Experiment Or Study Complete
6.5.20	Where The Experiment Or Study Shows That The Design Detail Is Invalid, Then Rework The Detail Development From 6.3.4	187-188	188 Design Detail Shown To Be Invalid
6.5.21	Where The Experiment Or Study Is Inconclusive, Reiterate From 6.5.18	187-189	189 Experiment Or Study Inconclusive
6.5.22	Where The Experiment Or Study Indicates That The Design Detail Is Supported, Reiterate From 6.5.17 Until All Design Details Based Up On Voluntarily Assigned Data And/or Simplifying Assumptions Are Validated	184, 187-190	190 All Design Details Based Upon Assumptions Now Validated
6.5.23	Taking The Goals Identified Under Section1.1 And 4.4.1 Together With The Constraints Listed Under Section 2.1 And The Performance Specification Set Out Under 5.1.20, Formulate Hypotheses Calculated To Facilitate Validation Of The Overall Design(s) Under 6.4.4	17, 66, 98, 134, 168-191	191 Hypotheses Formulated
6.5.24	For Each Hypothesis Under 6.5.23, Devise An Experiment To Test The Hypothesis And Validate The Design (See Note After 6.5.9)	191-192	192 Design For Experiment Or Study Ready
6.5.25	Assemble The Collection Of Proposed Experiments Or Studies Prepared Under 6.5.24 Into A Validation Program	192-193	193 Validation Program Ready
6.5.26	Distinguish Between Validation Experiments Or Studies To Be Carried Out Before Communication Of The Final Design Solution(s) And Those Which Must Be Carried Out On Manufactured Models Or Prototypes	193-194	194 Studies To Be Carried Out Before Communication Of The Final Design Solution(s) Identified
		193-195	195 Studies To Be Carried Out On Prototypes Identified
6.5.27	Select An Appropriate Validation Experiment Or Study From The List Of Those Which Are To Be Carried Out Before Communication Of The Final Design Solution(s), And Conduct Experiment	194-196	196 Experiment Or Study Complete
6.5.28	Where The Experiment Or Study Shows That The Design Is Invalid, Then Rework From 5.3.1	196-197	197 Design Shown To Be Invalid
6.5.29	Where The Experiment Or Study Indicates That The Design Is Supported, Reiterate From 6.5.23	196-198	198 Experiment Or Study Inconclusive
6.5.30	Where The Design Is Supported, Reiterate From 6.5.27 Until All Studies To Be Carried Out Before Communication Of The Final Design Solution(s) Are Complete	196-199	199 All Validation Studies To Be Carried Out Before Communication Of The Design Complete
6.5.31	Prepare A Report On Those Validation Experiments Or Studies Which Are To Be Carried Out On Models Or Prototypes	195-200	200 Report On Validation Studies To Be Carried Out On Manufactured Models Or Prototypes Now Ready

Phase 6



Item	Activity	Activity No	Event
7	**Communication**		
7.1	**Define Communication Needs**		
7.1.1	Referring To Instructions (See 0.2.1, And Any Subsequent Amendments At 1.0, 3.0, 5.0 And/or 6.0, Determine Addressee And Channel For Communication Of Design	199-201	201 Addressee And Channel For Communication Determined
7.1.2	Similarly Referring To Instructions, And Also To Course Of Action Adopted (See 0.4.6 And Subsequent Amendments At 2.2.9, 4.4.7 And 5.4.12), Determine Whether The General Spirit, The Geometry And Performance Or The Detailed Construction Of The Design Is To Be Communicated	199-202	202 Depth Of Detail For Communication Determined
7.1.3	Referring To The Definition Of The Design Idea (6.1.1) And The Statement Of The Range Of Variation Embraceable By The Design Idea (6.1.2) And Also To The Developed Overall Solution(s), Appraise What Is To Be Communicated	145, 146, 199-203	203 Matter For Communication Appraised
7.2	**Select Communication Medium**		
7.2.1	List All Communication Media Available	201, 202, 203-204	204 List Of Communication Media Ready
7.2.2	Referring To Communication Needs (Section 7.1), Select Suitable Media From The List Prepared Under 7.2.1	201, 203, 204-205	205 List Of Suitable Media Ready
7.2.3	Referring To Timetable (5.4.10) And Facilities (5.4.11), Select Medium To Be Used In Communicating The Design	138, 139, 205-206	206 Medium For Communication Selected
7.2.4	Reappraise And If Necessary Reformulate Course Of Action (Previously Set Out In 5.4.9)	137, 202, 206-207	207 Course Of Action Reformulated
7.3	**Prepare Communication**		
7.3.1	Referring To 7.1.3, List What Is To Be Communicated	203, 207-208	208 List Of Matter For Communication
7.3.2	Referring To Communication Needs (7.1.1 And 7.1.2), And In Terms Of The Medium (7.2.3), Describe Each Item Under 7.3.1	168, 202, 206, 208-209	209 Item Described
7.3.3	Prepare A Schedule Or Key Diagram Of All Items Described	208-210	210 Item Schedule For Key Ready
7.3.4	Prepare A Schedule Or Key Diagram Of All Documents Or Other Media Forming Part Of, Or Referred To In, The Communication	209, 210-211	211 Document Schedule Or Key Ready
7.3.5	Both Within Each Of, And Between, The Two Schedules 7.3.3 And 7.3.4, Index All Items So That Both The Correlation And The Tracing Of The Consequences Of Any Subsequent Revision Are Facilitated	210, 211-212	212 Schedules Cross Indexed
7.3.6	Check Each Document (Or Other Medium) For Any Deficiency In The Items Described	210, 211-213	213 Document Contents Checked
7.3.7	Check Each Document (Or Other Medium) For Any Deficiency In The Medium Itself	211-214	214 Document Quality Checked
7.3.8	Referring To Communication Needs (7.1), Examine Completed Communication	209, 210, 211, 212, 213, 214-215	215 Completed Communication Examined
7.3.9	Reappraise, And If Necessary Reformulate, Course Of Action (Reappraised At 7.2.4)	209, 215-216	216 Course Of Action Reformulated

Phase 6 (Continued)

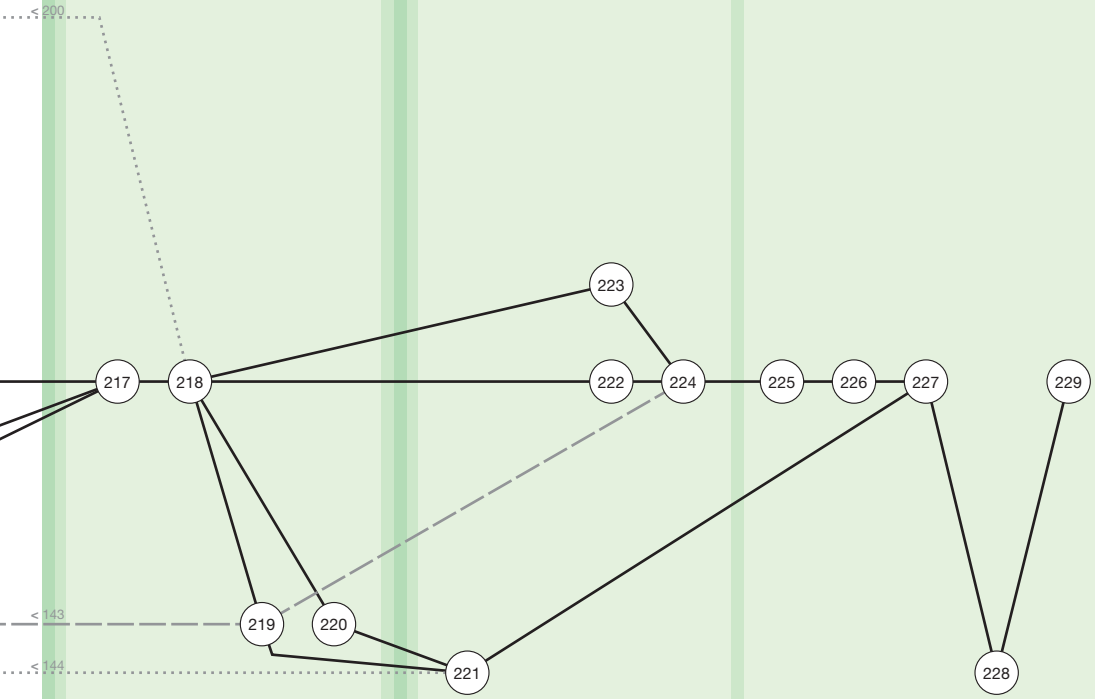
Communication (Continued)

Prepare Communication (Continued)

Winding Up

Wind Up Project

Close Records

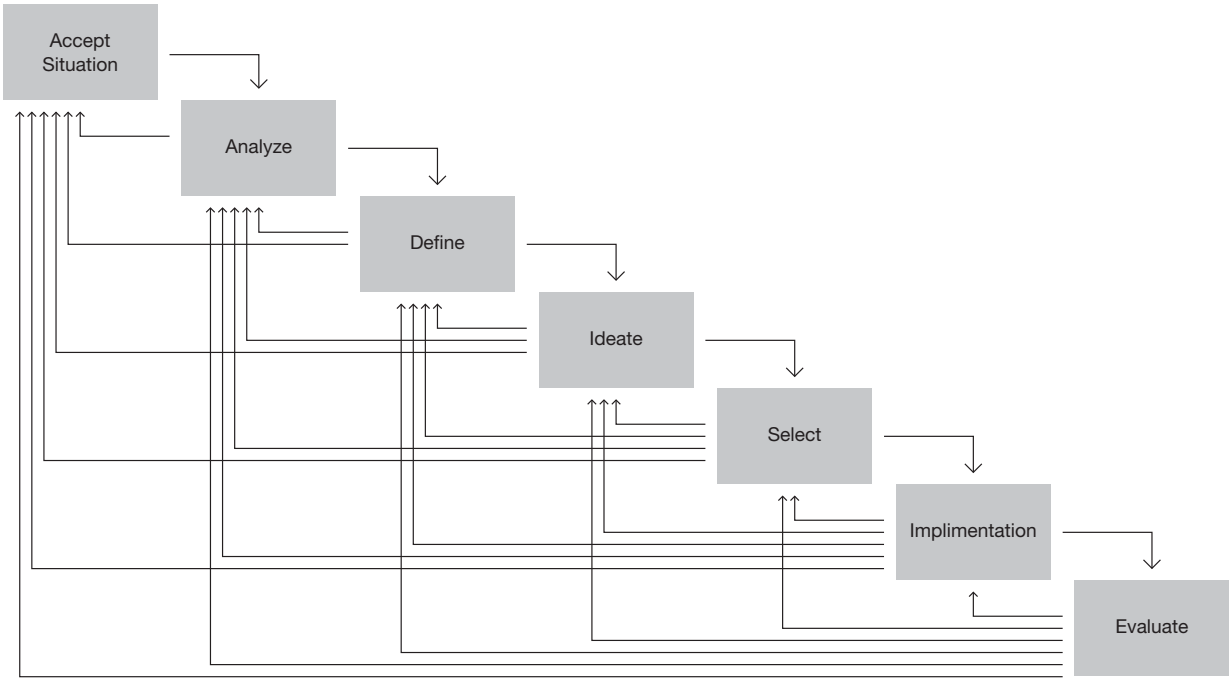


Item	Activity	Activity No	Event
7.4	**Transmit Information**	207, 210, 211, 216-217	217 Record Copies Ready
7.4.1	Make Security And/or Record Copies Of All Documents (Or Other Media)	200, 217-218	218 Communication Set Compiled
7.4.2	Compile The Communication Set		
7.4.3	Check For Completeness	218-219	219 Communication Set Dispatched
7.4.4	Enclose, Address And Dispatch The Communication	218-220	220 Advice Note Dispatched
7.4.5	Advise Dispatch Of Communication Through An Alternative Channel In The Case Of Sketch Designs Prepared Under 5.4.8, Continue Now From 5.4.9. In The Case Of Final Design Submissions, Await Authority To Terminate Project Or, Where Necessary, Reiterate From 5.3.1		
8	**Winding Up**		
8.1	**Wind Up Project**	219, 220-221	221 Communication Filed
8.1.1	Label And Store Copy Of The Communication As Dispatched	218-222	222 Auxiliary Transactions Complete
8.1.2	Complete Copyright Or Other Auxiliary Transactions	218-223	223 Financial Transactions And Book Keeping Complete
8.1.3	Complete Financial Transactions And Book Keeping	219, 222, 223-224	224 Correspondence Closed
8.1.4	Formally Discharge Obligations And Close Correspondence	224-225	225 Project Closed
8.1.5	Disengage From Problem And Close Project		
8.2	**Close Records**	225-226	226 Materials And Equipment Disposed Of
8.2.1	Store, Return, Dispose Of, Or Write Off All Remaining Materials And Equipment	212, 226-227	227 Redundant Material Disposed Of
8.2.2	Destroy All Obsolete And Transitory Material	227-228	228 Records Stored
8.2.3	Collate, Label And Store All Records	228-229	229 Appraisal Of Experience Complete
8.2.4	Appraise Experience Gained And Adjust Systems And Standards		

Seven-step process as a cascade with feedback
after Don Koberg and Jim Bagnall (1972)

In this model, Koberg and Bagnall have added feedback to their seven-stage model. (See page 16.) They note "one stage need not follow another . . . It is also possible that the stages can be considered in other ways . . . It could be circular . . . Others see it as a constant feedback system where you never go forward without always looping back to check on yourself; where one progresses by constant backward relationships; and where the stages of the process advance somewhat concurrently until some strong determining variable terminates the process (time, money, energy, etc.)"

Koberg and Bagnall go on to describe alternatives: viewing the design process as a branching system, and then as a "horse race" where each stage proceeds concurrently rather than a "mule train" where each stage proceeds one after the next. Finally, they note, "Process never ends . . . its ultimate model is the spiral, a continuum of sequential round trips that go on ad infinitum."



Cyclic models

We tend to think of a process in terms of steps—as a sequence.

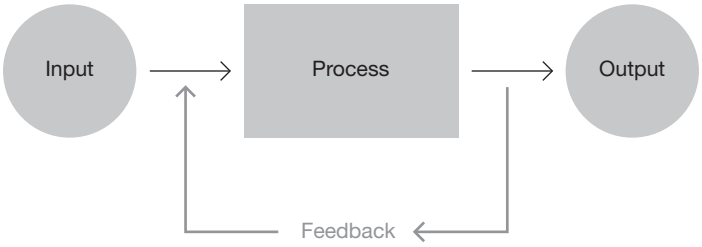
But designers require feedback, and most design processes include feedback loops.

In this section we examine models emphasizing feedback and continuous improvement.

Process with feedback (archetype)

This simple model recalls our first process model. (See page 12.) What's added is a feedback loop. More precisely, some of the output signal is split off and "fed back" into the input signal.

This happens all the time in design—at many levels. (See the previous spread.) We should be careful not to mistake this schematic diagram (or circuit diagram) for the actual design process. I include it here to underscore the importance of feedback in designing.

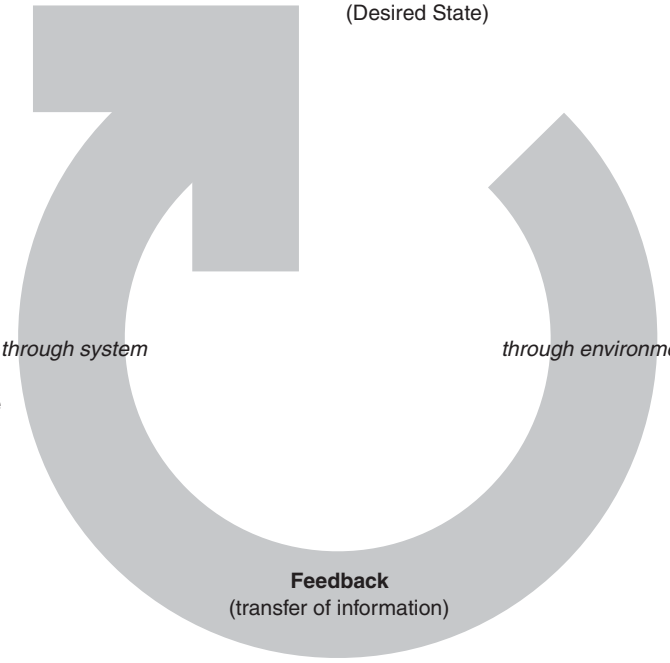


Goal-action-feedback loops
after Pangaro (2002)

Paul Pangaro describes feedback loops in terms of a goal-action-effect-measurement cycle. In this model, a system acts to accomplish a goal within its environment. The system measures the effect its actions have on the environment and compares the effect to its goal. Then the system looks for errors and acts (or re-acts) to correct them. By repeating the cycle, the system converges on a goal or maintains a steady state. Feedback is the information loop flowing from the system through the environment and back into the system. (For example, a boat pilot tacking to reach port or a thermostat turning a heater on and then off.)

Designers follow this cycle. They have goals, act to accomplish them, and measure their results to see if they meet their goals—goal-action-feedback. We've seen several analogs of this process—define-prototype-evaluate and design-build-test. (See pages 50-51.)

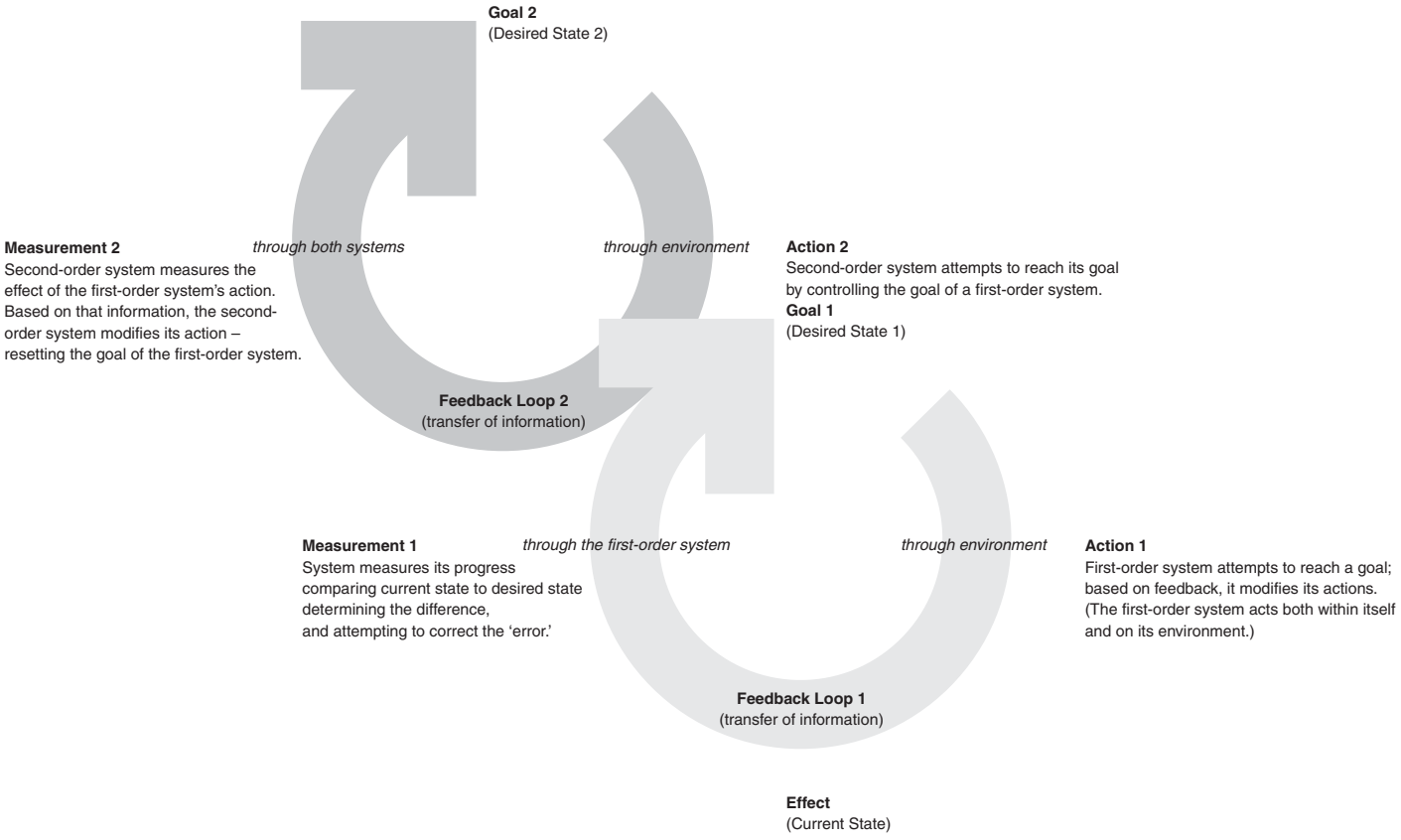
Feedback is a central subject of cybernetics, the science of goal-directed systems. Cybernetics has much to teach us about fundamental structures of design.



Second-order feedback loops
after Pangaro (2002)

The model on the previous page assumes a constant goal. That is, it provides no mechanism for changing or refining the system's goal. Typically, such systems are mechanical (or electronic) and require humans to set their goals. (For example, defining the set-point for a thermostat.) The human creates a second loop in which the "action" is setting the goal of the first loop. (Like the thermostat, the human also measures the room temperature and decides whether to raise or lower the set-point on the thermostat.)

As we've seen, designing involves not only achieving goals but also defining them. Thus we may improve our model of designing by nesting our original feedback loop within a second feedback loop. See the next page for an example.

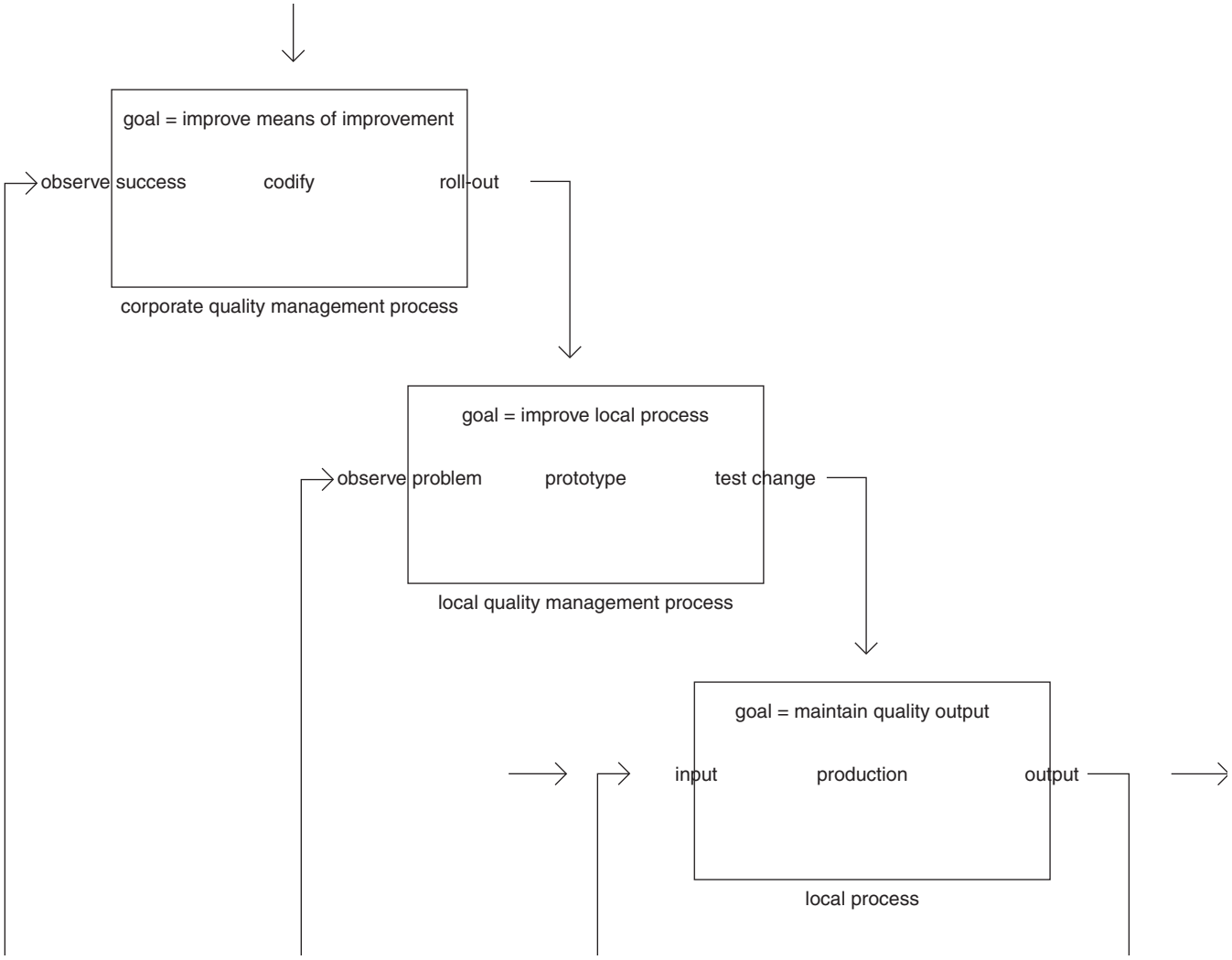


Bootstrapping or improving improvement
after Douglas Engelbart (1992)

In 1992, Douglas Engelbart offered “an optimized bootstrapping approach for drastically improving on any organization’s already existing improvement processes.”

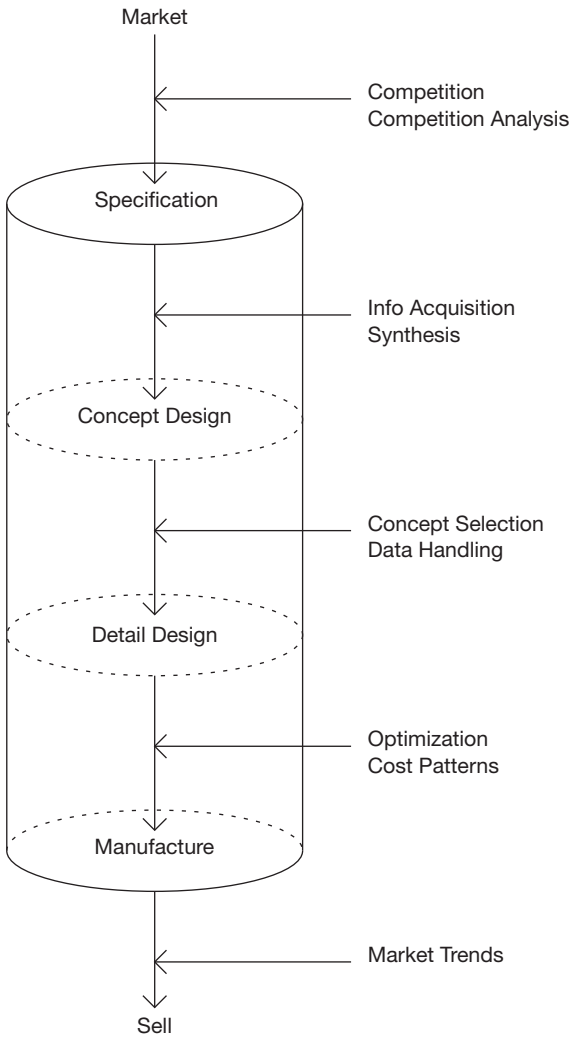
According to his foundation, Bootstrap.org, the process works as follows, “Referring to an organization’s principal work as an A-activity and to ordinary efforts at process improvement as a B-activity, he denotes bootstrapping as a C-activity, which is an improving of the improvement process. His paper ‘Toward High-Performance Organizations: A Strategic Role for Groupware’ argues that highest payoff comes from engaging in that C-activity.”

Levels A, B, and C are analogous to first-, second-, and third-order feedback loops.



Product development process
after Stuart Pugh (1990)

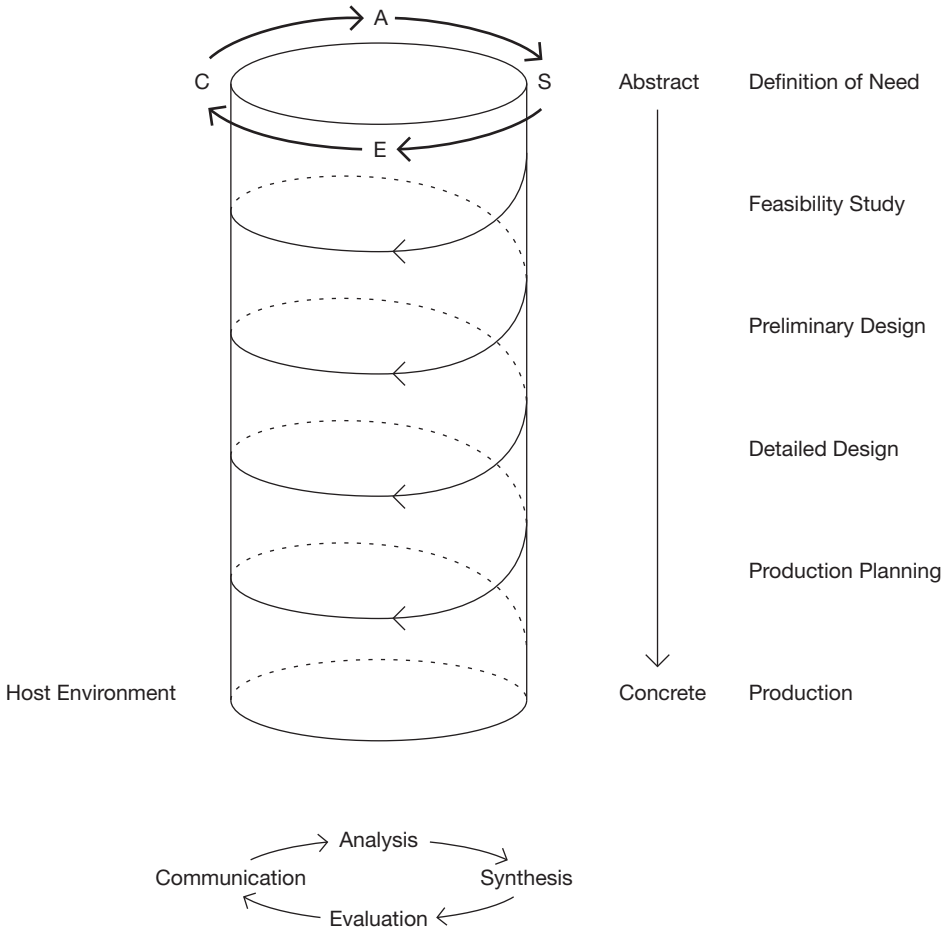
Pugh published this model in his book, *Total Design: Integrated Methods for Successful Product Engineering*. His reasons for presenting it as a cylinder are not clear from the diagram itself.



Iconic model of the design process
after Mihajlo D. Mesarovic (1964)

In this model, Mesarovic employs a helix as the central structure, suggesting both a repeated cycle of steps and progress through time.

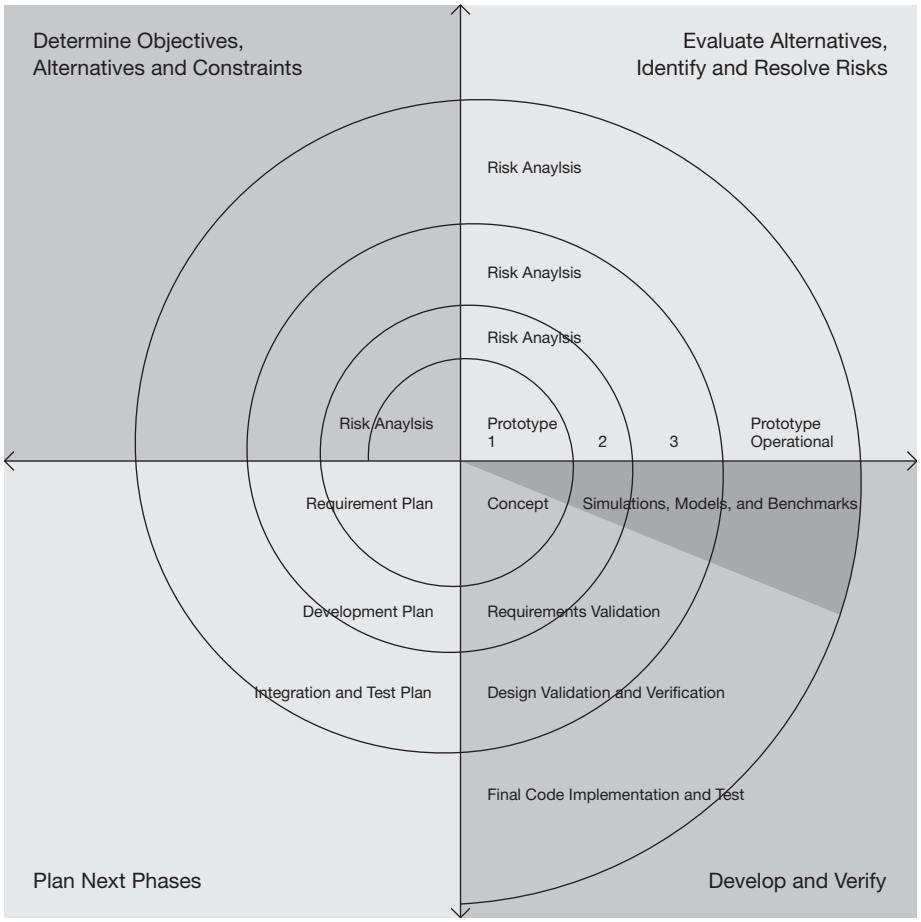
Peter Rowe (1987) notes that Mesarovic's model is similar in structure to Asimow's. (See pages 92-95.) "Throughout this kind of account runs the assumption that it is possible to discriminate distinct phases of activity and, further more, that such distinctions have relevance to our understanding of design." Rowe continues, "The very maintenance of distinct phases of activity, with a beginning and an end, and with feedback loops among them, requires that objective performance criteria can be explicitly stated in a manner that fundamentally guides the procedure. Moreover, there is a strong implication that the eventual synthesis of information in the form of some designed object follows in a straightforward fashion from analysis of the problem at hand together with likely performance criteria. Therefore, once a problem has been defined, its solution is made directly accessible in terms of that definition." Rowe describes this view as "behaviorist" and also links it to "operations research".



Spiral model of software development
after Barry Boehm (1986)

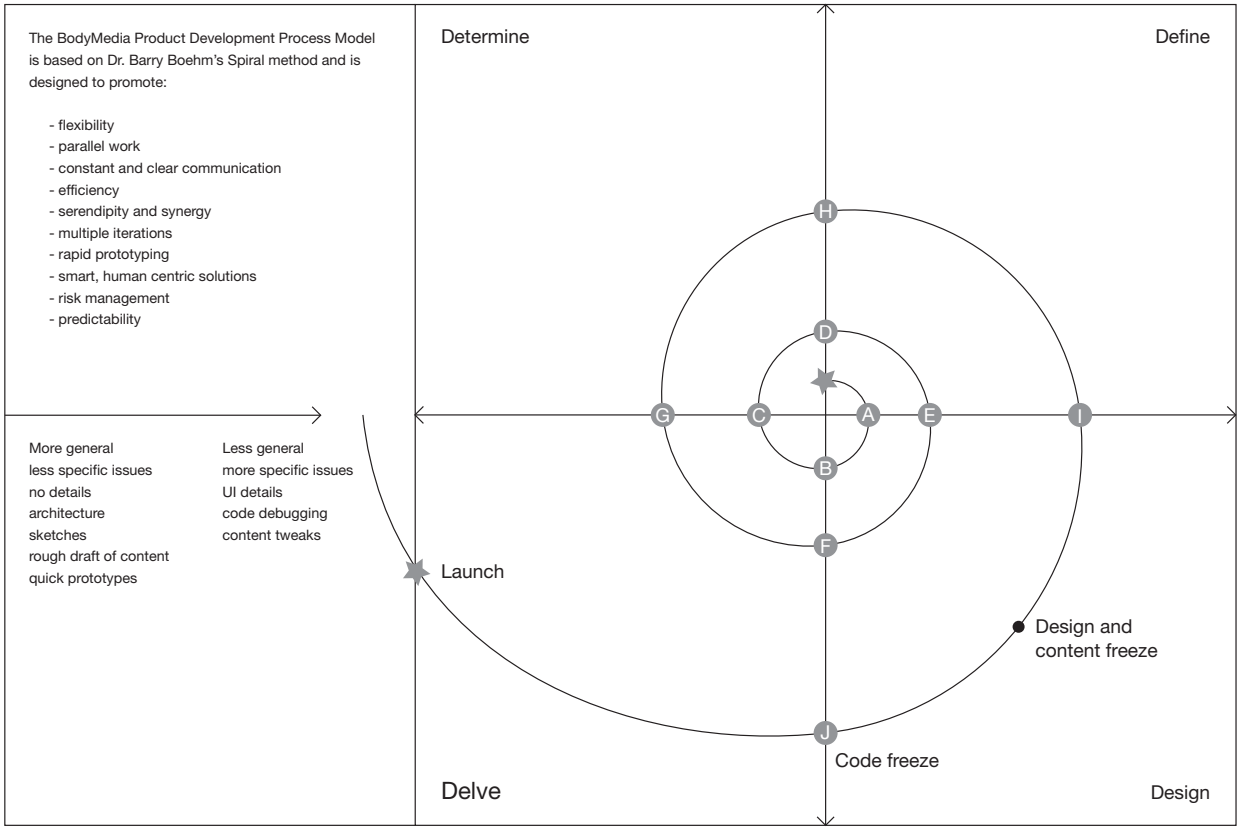
Boehm represented repeating cycles of design with a spiral path moving away from a center starting point.

In addition to the spiral shape, Boehm introduces a focus on risk reduction. Gary Schmidt of Washburn University offers this description of Boehm's model, "The radial dimension of the model represents the cumulative costs when finishing the steps. The angular dimension represents the progress made in completing each cycle. Each loop of the spiral from x-axis clockwise through 360 represents one phase. One phase is split roughly into four sectors of major activities
- Objective setting
- Risk assessment and reduction
- Development and validation
- Planning the next phases"



BodyMedia product development process
after Chris Pacione (2002)

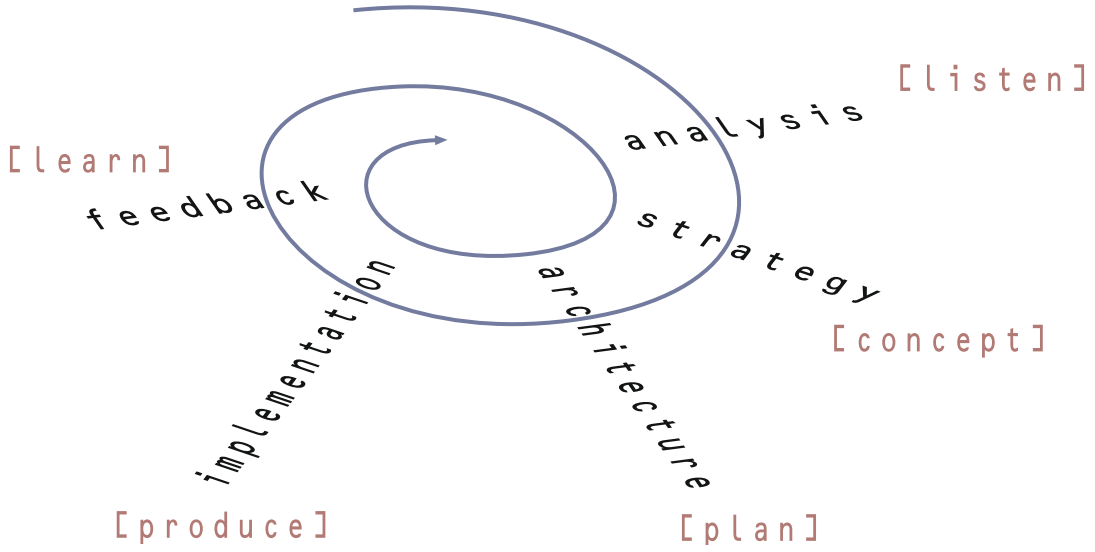
In his book, *What Is Web Design?*, Nico MacDonald (2003) published a similar model Pacione developed for his company, BodyMedia. MacDonald notes, “The model requires that the product must be the right thing to make, posits designers as synthesizers and indicates the relationship with users is on-going.” Note also Pacione’s variation on the 4Ds—in this case define-design-delve-determine. (See page 62.)



★ — A	Define-	Brainstorm and Kickoff
A — B	Design-	the Information Architecture and quick interactive prototypes
B — C	Delve-	into prototype and test the usefullness of the idea
C — D	Determine-	the weakness, strengths and risks, time constraints
D — E	Define-	new alternatives, and solutions
E — F	Design-	update architecture, content, code and UI of interactive portotype
F — G	Delve-	into refined solution, test for usability, and major bugs
G — H	Determine-	solutions weaknesses, strenghts, risks, time constraints
H — I	Define-	what needs to (can) change with code, UI
I — J	Design-	Design and content freezes, all specs due. Code freeze
J — ★	Final testing and Q&A, Launch!	

Design process
after Paul Souza (1996)

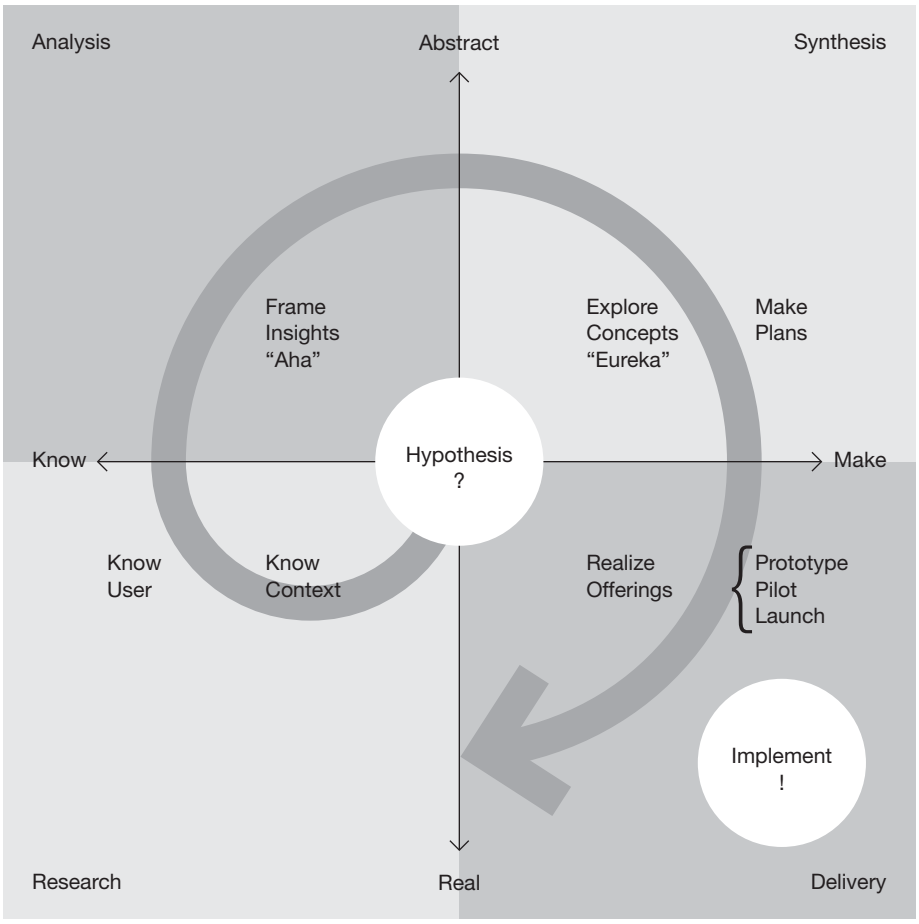
Souza also used a spiral path to represent repeating cycles in the design process. In Boehm's model, the spiral travels out from the center suggesting—though perhaps not intentionally—that the process diverges. Traveling outward could also suggest adding increasing amounts of detail. In Souza's model, the path travels in toward the center suggesting the process converges on a goal.



Innovation planning
after Vijay Kumar (2003)

Kumar presented this model at the 2003 HITS Conference (Humans, Interaction, Technology, Strategy) in Chicago. He described modes of planning (rather than steps) emphasizing the iterative and interconnected nature of the design process. He has also mapped tools and methods onto each of the modes. He spoke of innovation as the jump from insight to concept—from aha to eureka—describing it as a revelation, magic, genius, intuition, a hunch.

Kumar teaches at the Illinois Institute of Technology's Institute of Design; his teaching and research includes a focus on understanding innovation.

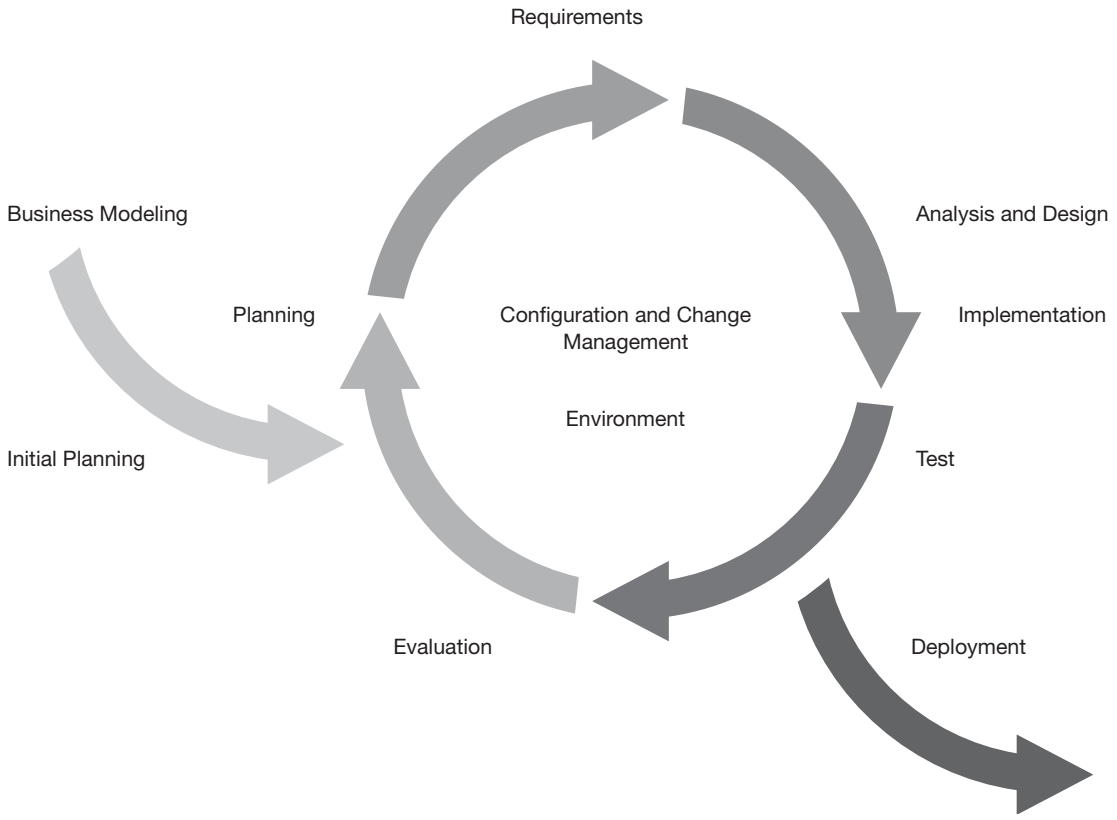


Rational Unified Process iteration cycle
after Per Kroll (2004)

Iteration is a central principle of the Rational unified process. Kroll notes, “Each iteration includes some or most of the development disciplines (requirements, analysis, design, implementation and [testing activities]. Each iteration also has a well-defined set of objectives and produces a partial working implementation of the final system. And each successive iteration builds on the work of previous iterations to evolve and refine the system until the final product is complete. Early iterations emphasize requirements as well as analysis and design; later iterations emphasize implementation and testing.”

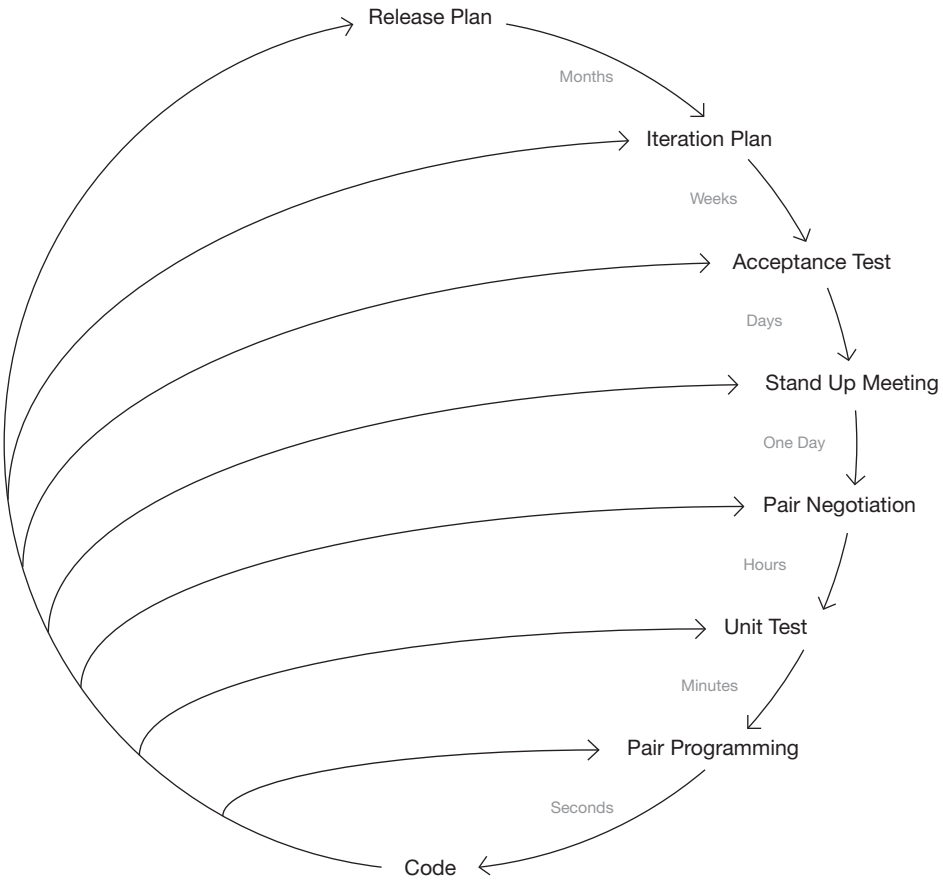
Knoll suggests four principles:
1. Build functional prototypes early.
2. Divide the detailed design, implementation and test phases into iterations.
3. Baseline an executable architecture early on.
4. Adopt an iterative and risk-driven management process.

Kroll is director of the Rational Unified Process development and product management teams at IBM. (For another model of RUP, see page 67.)



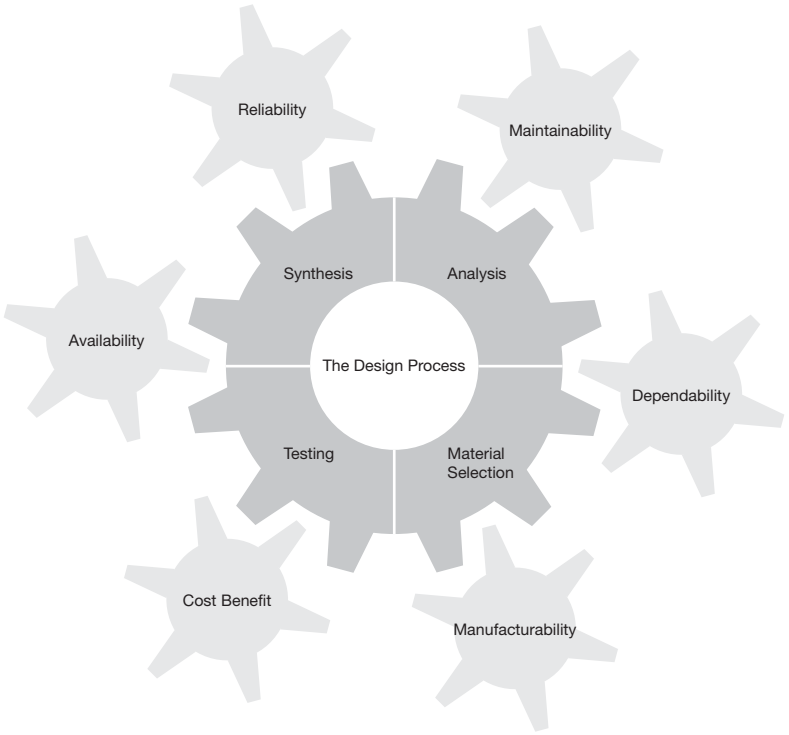
Extreme programming planning/feedback loops
after Don Wells (2000)

Extremeprogramming.org published this hypertext model depicting nested feedback loops within the XP development process. The length of each loop increases from bottom to top of the model. The model also serves as a sort of table of contents for key ideas in extreme programming. (For other models of extreme programming, see pages 70-71.)



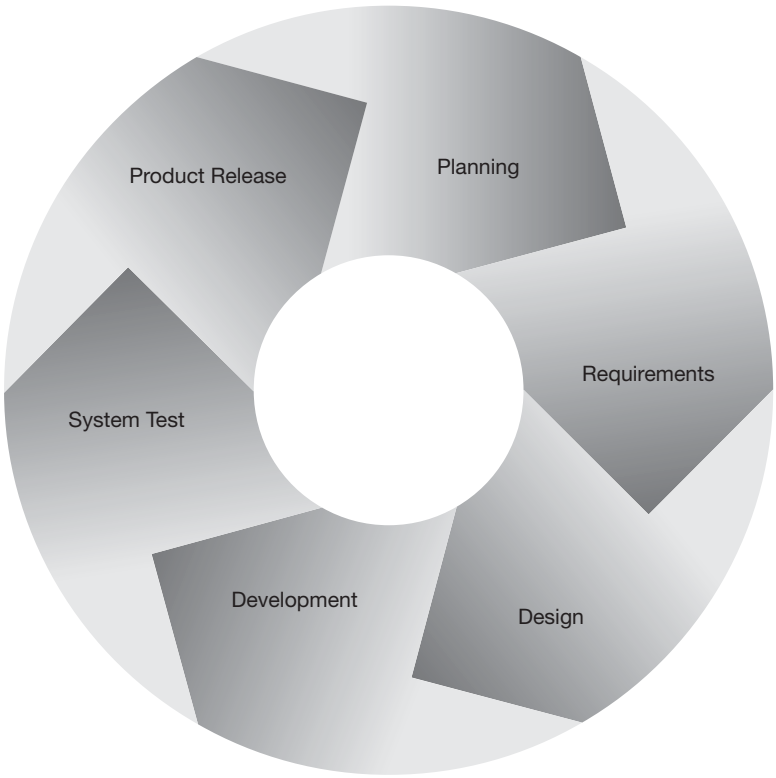
Engineering design process
after Atila Ertas and Jesse C. Jones (1996)

This model is interesting in its use of the gear metaphor. Did the authors intend to frame design as a mechanical process? It's also unusual to see a model begin with synthesis—or include "materials selection" at the same level of abstraction. The inclusion of the six considerations or constraints is also unusual.



Product development process: overview
after Hewlett Packard (circa 2000)

Loops or circular layouts are curiously rare in design process models—with the notable exception of the PDCA cycle on the next page. Koberg and Bagnall provide another example by simply turning their seven-step process into a circle.



Our Focus:
– User analysis, requirements
– Product definition, design, and development for ease of use and usefulness

Other Elements of the Customer Experience:
– Ordering, delivery
– Documentation
– Installation
– Integration with 3rd party products
– Customer support

PDCA quality cycle
after Walter A. Shewart (1939)

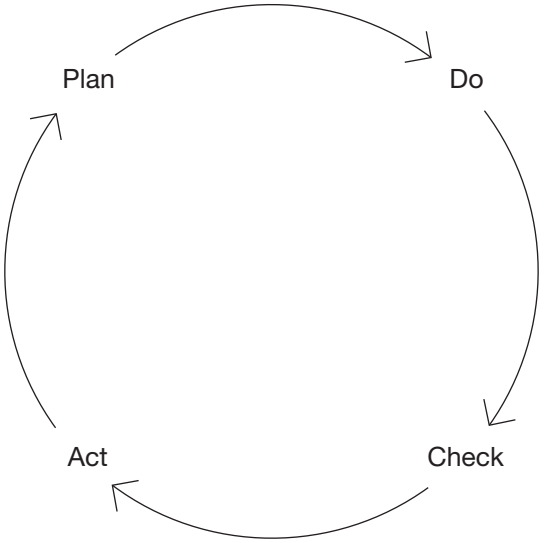
PDCA stands for plan-do-check-act cycle of continuous improvement, a standard principle of quality assurance and management. It is also known as the Shewhart cycle or the Deming cycle.

The mathematician Walter A. Shewhart was concerned with what he called “the formulation of a scientific basis for securing economic control.” In 1939, he published *Statistical Method from the Viewpoint of Quality Control*, the first place he discussed the PDCA concept, according to the American Society for Quality (ASQ).

Edward Deming worked with Shewhart at Bell Laboratories and later popularized the PDCA cycle, especially in Japan. Deming substituted “study” for “check”. PDCA and PDSA have many incarnations and many definitions. For example, the ISO 9001 standard includes the PDCA cycle. Over the last 20 years, the focus of quality management has expanded from manufacturing processes to include a systemic view of customer satisfaction.

Determine the root cause of the problem
then plan a change or a test
aimed at improvement.

Carry out the change or the test,
preferably in a pilot or on a small scale.



Adopt the change,
if the desired result was achieved.
If the result was not desired,
repeat the cycle using knowledge obtained

Check if the desired result was achieved,
what, if anything, went wrong,
and what was learned.

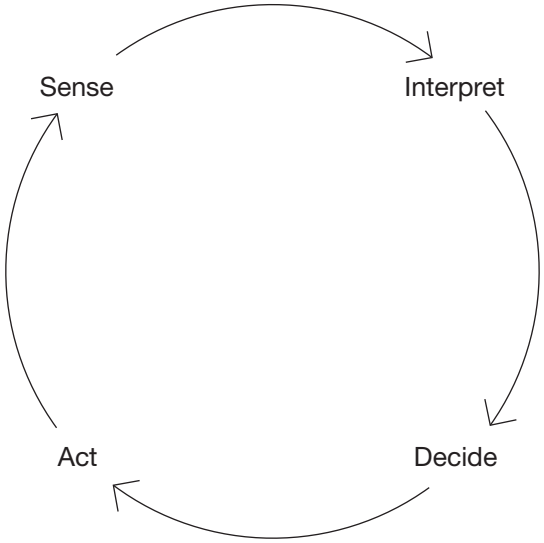
Adaptability loop
after Stephan H. Haeckel (2003)

Haeckel proposed this process for managing within a changing environment. At first, it appears to be a classic feedback-based control loop. But the options for action include changing goals and thus suggest a more complex process than is represented in the model.

Haeckel's model may also be interpreted as a variation on the classic PDCA cycle. It's interesting to note that the PDCA cycle also implies but does not represent a process for changing goals. (Some variations on the model include it.) The authors may have chosen a simpler representation to make it easy to communicate and remember.

Deploys probes and gather a wide range of signals, from hard data to financail reports to conversations with customers, to spot impending change in your business environment, your customer's businesses and their customer's businesses. Internally, track breakdowns in your organization's performance, and violations of commitments and basic rules. Scan for capabilities firms in other industries have that you might find useful. All data should be viewable on PCs and handhelds.

Use a variety of technologies and technique to understand the meaning of what your probes are sensing, and why commitments between roles break down. For example, scenario planning can help identify and prepare for ways the future may unfold, war-gaming helps leaders think through various responses to change, and collaborative decision-making helps leaders decide how to allocate limited resources in unpredictable environments. This information should be viewable on screen.



As leader, publicly proclaim the new *raison d'être*, basic rules and roles, or—if no change is required at the time—affirm the old ones. When changing any roles, make sure the right people execute them and that their commitments to others within the organization are adjusted to reflect any new desired outcomes, Likewise, install or upgrade electronic information systems to display the updated commitments as well as any new "sense" and "interpret" information now required.

Should organization's *raison d'être*, policies or role and accountability design change? (For example, if commitments are repeatedly broken, if new capabilities aren't being incorporated, etc.) Frequently challenged policies might need to be relaxed, tightened or eliminated. The *raison d'être* might need changing if there are big or sudden shifts in the marketplace or in the values and preferences of your most important customers and other constituents.

Complete list of models

Introducing process

10
Design Process
after Tim Brennan (~1990)

12
Process archetype

13
On the infinite expandability of process models

14
Design process archetype: Analysis, Synthesis
after Koberg and Bagnall (1972)

15
Problem, Solution
after JJ Foreman (1967)

16
Expanding the two-step process
after Don Koberg and Jim Bagnall (1972)

17
Matching process to project complexity
after Jay Doblin (1987)

18
Unself-conscious and self-conscious design
after Christopher Alexander (1962)

Analysis synthesis evaluation

20
Oscillation

21
Programming and designing
after William M. Pena and Steven A. Parshall (1969)

22
Diverge / Converge vs Narrow / Expand

23
Decomposition / recombination
after VDI 2221 (from Cross 1990)

24
Dynamics of divergence and convergence
after Bela H. Banathy (1996)

25
Overall, the design process must converge
after Nigel Cross (2000)

26
Gradual shift of focus from analysis to synthesis
after Bill Newkirk (1981)

27
Problem to solution: sequence or parallel process or loop?

Academic models

28
Walking process
after Lawson (1980)

30
Four-stage design process
after Nigel Cross (2000)

31
Engineering design process
after Michael J. French (1985)

32
VDI 2221: System Approach to the Design of Technical Systems and Products
after Verein Deutscher Ingenieure (1987)

33
Design process
after Gerhard Pahl and Wolfgang Beitz (1984)

34
Architect's Plan of Work (schematic)
after the Royal Institute of British Architects Handbook (1965)

35
Architect's Plan of Work, (detailed)
after the RIBA Handbook (1965)

36
Problem solving process
after George Polya (1945)

37
Scientific problem solving process
after Cal Briggs and Spencer W. Havlick (1976)

38
THEOC, a model of the scientific method

39
Criteria of validation of scientific explanations (CVSE)
after Humberto Maturana (1987)

40
Comprehensive anticipatory design science
after Buckminster Fuller (1978?)

41
Design Process and Practice
after Richard Buchannan (1997)

42
Creative process
after Bryan Lawson (1980)

43
Primary generator
after Jane Darke (1978)

44
Design process
after Jane Darke (1978)

45
Design process
after Thomas A. Marcus (1969) and Thomas W Maver (1970)

47
Process of designing solutions
after Clement Mok and Keith Yamashita (2003)

48
Case study, using the AIGA process in Iraq
by Nathan Felde (2003)

49
What the AIGA didn't tell you
by Nathan Felde (2003)

51
Design, build, test (1 of 3)
after Alice Agogino

52
Design, build, test (2 of 3)
after Alice Agogino

53
Design, build, test (3 of 3)
after Alice Agogino

54
Mechanical engineering design process
after students at UC Berkeley Institute of Design (BID)

55
New product development process
after Steven D. Eppinger and Karl T. Ulrich (1995)

57
Design Process
after John Chris Jones (1970)

58
Value analysis
after John Chris Jones (1970)

59
Man-machine system designing
after John Chris Jones (1970)

Consultant models

60
Eight phases of a project
Sometimes presented as six phases of a project

62
4D software process
and variations

63
IT consulting process overview
after Mindtree Consulting

65
IDEO
(2004)

66
Trees

Software development models

68
Waterfall lifecycle
after Philippe Kruchten (2004)

69
Rational Unified Process (RUP)
after Phillippe Kruchten (2003)

Complete list of models
continued

70
Extreme Programming (XP) Process
after Don Wells (2000)

72
V model
Paul Rock (~1980), IABG (1992)

73
Joint Application Development (JAD)
after Jane Wood and Denise Silver (1995)

74
PMBOK (Project Management Body of Knowledge)
PMI (Project Management Institute) (1987)

75
ISO 13407 Human-centered design processes for interactive systems
Tom Stewart et al. (1999)

76
User-centered design process (UCD)
after Karel Vredenburg (2003)

77
Relation of UCD to IPD and Business Management
after Karel Vredenburg (2003)

78
Sun Sigma Framework
DMADV methodology for new products

79
Sun Sigma Framework
DMAIC methodology for improving existing products

80
Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)
“Mapped” processes for product instances

81
Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)
“Mapped” processes for product lines

Complex linear models

84
Web development process
after Vanguard Group (circa 1999)

87
Evolution of the software development process
after Alan Cooper (2001)

88
Goal-directed design process
after Alan Cooper (2001)

90
Idealized process of developing buildings
after Alan Cooper (2004)

91
Idealized process of developing software
after Alan Cooper (2004)

93
Morphology of design
after Morris Asimow (1962)

97
Biological sequence of problem solving
after Bruce Archer (1963-1964)

98
Basic design procedure
after Bruce Archer (1963-1964)

99
Check list for product designers
Bruce Archer (1963-1964)

Cyclic models

114
Seven-step process as a cascade with feedback
after Don Koberg and Jim Bagnall (1972)

116
Process with feedback (archetype)

117
Goal-action-feedback loops
after Pangaro (2002)

118
Second-order feedback loops
after Pangaro (2002)

119
Bootstrapping or improving improvement
after Douglas Engelbart (1992)

120
Product development process
after Stuart Pugh (1990)

121
Iconic model of the design process
after Mihajlo D. Mesarovic (1964)

122
Spiral model of software development
after Barry Boehm (1986)

123
BodyMedia product development process
after Chris Pacione (2002)

124
Design process
Paul Souza (1999?)

125
Innovation planning
after Vijay Kumar (2003)

126
Rational Unified Process iteration cycle
Per Kroll (2004)

127
Extreme programming planning/feedback loops
after Don Wells (2000)

128
Engineering design process
after Atila Ertas and Jesse C. Jones (1996)

129
Product development process: overview
Hewlett Packard (circa 2000)

130
PDCA quality cycle
after Walter A. Shewart (1939)

131
Adaptability loop
after Stephan H. Haeckel (2003)

Chronological list

130
PDCA quality cycle
after Walter A. Shewart (1939)

36
Problem solving process
after George Polya (1945)

18
Unself-conscious and self-conscious design
after Christopher Alexander (1962)

93
Morphology of design
after Morris Asimow (1962)

97
Biological sequence of problem solving
after Bruce Archer (1963-1964)

98
Basic design procedure
after Bruce Archer (1963-1964)

99
Check list for product designers
Bruce Archer (1963-1964)

121
Iconic model of the design process
after Mihajlo D. Mesarovic (1964)

34
Architect's Plan of Work (schematic)
after the Royal Institute of British Architects Handbook (1965)

35
Architect's Plan of Work, (detailed)
after the RIBA Handbook (1965)

15
Problem, Solution
after JJ Foreman (1967)

45
Design process
after Thomas A. Marcus (1969) and Thomas W Maver (1970)

21
Programming and designing
after William M. Pena and Steven A. Parshall (1969)

57
Design Process
after John Chris Jones (1970)

58
Value analysis
after John Chris Jones (1970)

59
Man-machine system designing
after John Chris Jones (1970)

14
Design process archetype: Analysis, Synthesis
after Koberg and Bagnall (1972)

16
Expanding the two-step process
after Don Koberg and Jim Bagnall (1972)

114
Seven-step process as a cascade with feedback
after Don Koberg and Jim Bagnall (1972)

37
Scientific problem solving process
after Cal Briggs and Spencer W. Havlick (1976)

43
Primary generator
after Jane Darke (1978)

44
Design process
after Jane Darke (1978)

40
Comprehensive anticipatory design science
after Buckminster Fuller (1978?)

28
Walking process
after Lawson (1980)

42
Creative process
after Bryan Lawson (1980)

26
Gradual shift of focus from analysis to synthesis
after Bill Newkirk (1981)

33
Design process
after Gerhard Pahl and Wolfgang Beitz (1984)

31
Engineering design process
after Michael J. French (1985)

122
Spiral model of software development
after Barry Boehm (1986)

17
Matching process to project complexity
after Jay Doblin (1987)

39
Criteria of validation of scientific explanations (CVSE)
after Humberto Maturana (1987)

74
PMBOK (Project Management Body of Knowledge)
PMI (Project Management Institute) (1987)

32
VDI 2221: System Approach to the Design of Technical Systems and Products
after Verein Deutscher Ingenieure (1987)

10
Design Process
after Tim Brennan (~1990)

23
Decomposition / recombination
after VDI 2221 (from Cross 1990)

120
Product development process
after Stuart Pugh (1990)

119
Bootstrapping or improving improvement
after Douglas Engelbart (1992)

72
V model
Paul Rock (~1980), IABG (1992)

55
New product development process
after Steven D. Eppinger and Karl T. Ulrich (1995)

73
Joint Application Development (JAD)
after Jane Wood and Denise Silver (1995)

24
Dynamics of divergence and convergence
after Bela H. Banathy (1996)

128
Engineering design process
after Atila Ertas and Jesse C. Jones (1996)

41
Design Process and Practice
after Richard Buchannan (1997)

124
Design process
Paul Souza (1999?)

75
ISO 13407 Human-centered design processes for interactive systems
Tom Stewart et al. (1999)

84
Web development process
after Vanguard Group (circa 1999)

129
Product development process: overview
Hewlett Packard (circa 2000)

25
Overall, the design process must converge
after Nigel Cross (2000)

30
Four-stage design process
after Nigel Cross (2000)

70
Extreme Programming (XP) Process
after Don Wells (2000)

127
Extreme programming planning/feedback loops
after Don Wells (2000)

87
Evolution of the software development process
after Alan Cooper (2001)

Chronological list
continued

88
Goal-directed design process
after Alan Cooper (2001)

123
BodyMedia product development process
after Chris Pacione (2002)

117
Goal-action-feedback loops
after Pangaro (2002)

118
Second-order feedback loops
after Pangaro (2002)

47
Process of designing solutions
after Clement Mok and Keith Yamashita (2003)

48
Case study, using the AIGA process in Iraq
by Nathan Felde (2003)

49
What the AIGA didn't tell you
by Nathan Felde (2003)

131
Adaptability loop
after Stephan H. Haeckel (2003)

69
Rational Unified Process (RUP)
after Phillippe Kruchten (2003)

125
Innovation planning
after Vijay Kumar (2003)

76
User-centered design process (UCD)
after Karel Vredenburg (2003)

77
Relation of UCD to IPD and Business Management
after Karel Vredenburg (2003)

90
Idealized process of developing buildings
after Alan Cooper (2004)

91
Idealized process of developing software
after Alan Cooper (2004)

65
IDEO
(2004)

126
Rational Unified Process iteration cycle
Per Kroll (2004)

68
Waterfall lifecycle
after Philippe Kruchten (2004)

Dates not found

51
Design, build, test (1 of 3)
after Alice Agogino
52
Design, build, test (2 of 3)
after Alice Agogino

53
Design, build, test (3 of 3)
after Alice Agogino

54
Mechanical engineering design process
after students at UC Berkeley Institute of Design (BID)

60
Eight phases of a project
Sometimes presented as six phases of a project

62
4D software process
and variations

63
IT consulting process overview
after Mindtree Consulting

66
Trees

78
Sun Sigma Framework
DMADV methodology for new products

79
Sun Sigma Framework
DMAIC methodology for improving existing products

80
Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)
“Mapped” processes for product instances

81
Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)
“Mapped” processes for product lines

Produced for this book (2004)

116
Process with feedback (archetype)

12
Process archetype

13
On the infinite expandability of process models

19
*Analysis synthesis evaluation

20
Oscillation

22
Diverge / Converge vs Narrow / Expand

27
Problem to solution: sequence or parallel process or loop?

38
THEOC, a model of the scientific method

Author list

51
Design, build, test (1 of 3)
after Alice Agogino

52
Design, build, test (2 of 3)
after Alice Agogino

53
Design, build, test (3 of 3)
after Alice Agogino

18
Unself-conscious and self-conscious design
after Christopher Alexander (1962)

97
Biological sequence of problem solving
after Bruce Archer (1963-1964)

98
Basic design procedure
after Bruce Archer (1963-1964)

99
Check list for product designers
Bruce Archer (1963-1964)

93
Morphology of design
after Morris Asimow (1962)

24
Dynamics of divergence and convergence
after Bela H. Banathy (1996)

122
Spiral model of software development
after Barry Boehm (1986)

10
Design Process
after Tim Brennan (~1990)

37
Scientific problem solving process
after Cal Briggs and Spencer W. Havlick (1976)

41
Design Process and Practice
after Richard Buchannan (1997)

87
Evolution of the software development process
after Alan Cooper (2001)

88
Goal-directed design process
after Alan Cooper (2001)

90
Idealized process of developing buildings
after Alan Cooper (2004)

91
Idealized process of developing software
after Alan Cooper (2004)

25
Overall, the design process must converge
after Nigel Cross (2000)

30
Four-stage design process
after Nigel Cross (2000)

43
Primary generator
after Jane Darke (1978)

44
Design process
after Jane Darke (1978)

17
Matching process to project complexity
after Jay Doblin (1987)

119
Bootstrapping or improving improvement
after Douglas Engelbart (1992)

55
New product development process
after Steven D. Eppinger and Karl T. Ulrich (1995)

128
Engineering design process
after Atila Ertas and Jesse C. Jones (1996)

48
Case study, using the AIGA process in Iraq
by Nathan Felde (2003)

49
What the AIGA didn't tell you
by Nathan Felde (2003)

131
Adaptability loop
after Stephan H. Haeckel (2003)

15
Problem, Solution
after JJ Foreman (1967)

31
Engineering design process
after Michael J. French (1985)

40
Comprehensive anticipatory design science
after Buckminster Fuller (1978?)

129
Product development process: overview
Hewlett Packard (circa 2000)

65
IDEO
(2004)

57
Design Process
after John Chris Jones (1970)

58
Value analysis
after John Chris Jones (1970)

59
Man-machine system designing
after John Chris Jones (1970)

14
Design process archetype: Analysis, Synthesis
after Koberg and Bagnall (1972)

16
Expanding the two-step process
after Don Koberg and Jim Bagnall (1972)

114
Seven-step process as a cascade with feedback
after Don Koberg and Jim Bagnall (1972)

126
Rational Unified Process iteration cycle
Per Kroll (2004)

69
Rational Unified Process (RUP)
after Phillippe Kruchten (2003)

68
Waterfall lifecycle
after Philippe Kruchten (2004)

125
Innovation planning
after Vijay Kumar (2003)

28
Walking process
after Lawson (1980)

42
Creative process
after Bryan Lawson (1980)

45
Design process
after Thomas A. Marcus (1969) and Thomas W Maver (1970)

39
Criteria of validation of scientific explanations (CVSE)
after Humberto Maturana (1987)

121
Iconic model of the design process
after Mihajlo D. Mesarovic (1964)

63
IT consulting process overview
after Mindtree Consulting

47
Process of designing solutions
after Clement Mok and Keith Yamashita (2003)

26
Gradual shift of focus from analysis to synthesis
after Bill Newkirk (1981)

123
BodyMedia product development process
after Chris Pacione (2002)

Author list
continued

33
Design process
after Gerhard Pahl and Wolfgang Beitz (1984)

117
Goal-action-feedback loops
after Pangaro (2002)

118
Second-order feedback loops
after Pangaro (2002)

21
Programming and designing
after William M. Pena and Steven A. Parshall (1969)

74
PMBOK (Project Management Body of Knowledge)
PMI (Project Management Institute) (1987)

36
Problem solving process
after George Polya (1945)

120
Product development process
after Stuart Pugh (1990)

34
Architect's Plan of Work (schematic)
after the Royal Institute of British Architects Handbook (1965)

35
Architect's Plan of Work, (detailed)
after the RIBA Handbook (1965)

72
V model
Paul Rock (~1980), IABG (1992)

130
PDCA quality cycle
after Walter A. Shewart (1939)

124
Design process
Paul Souza (1999?)

75
ISO 13407 Human-centered design processes for interactive systems
Tom Stewart et al. (1999)

78
Sun Sigma Framework
DMADV methodology for new products

79
Sun Sigma Framework
DMAIC methodology for improving existing products

80
Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)
"Mapped" processes for product instances

81
Sun Product Lifecycle (PLC)
Sun Software Development Framework (SDF)
"Mapped" processes for product lines

84
Web development process
after Vanguard Group (circa 1999)

76
User-centered design process (UCD)
after Karel Vredenburg (2003)

77
Relation of UCD to IPD and Business Management
after Karel Vredenburg (2003)

32
VDI 2221: System Approach to the Design of Technical Systems and Products
after Verein Deutscher Ingenieure (1987)

23
Decomposition / recombination
after VDI 2221 (from Cross 1990)

70
Extreme Programming (XP) Process
after Don Wells (2000)

127
Extreme programming planning/feedback loops
after Don Wells (2000)

73
Joint Application Development (JAD)
after Jane Wood and Denise Silver (1995)

Authors not identified

54
Mechanical engineering design process
after students at UC Berkeley Institute of Design (BID)

60
Eight phases of a project
Sometimes presented as six phases of a project

62
4D software process
and variations

66
Trees

Produced for this book (2004)

116
Process with feedback (archetype)

12
Process archetype

13
On the infinite expandability of process models

19
*Analysis synthesis evaluation

20
Oscillation

22
Diverge / Converge vs Narrow / Expand

27
Problem to solution: sequence or parallel process or loop?

38
THEOC, a model of the scientific method

Bibliography

Alexander, Christopher.
Notes on the Synthesis of Form
MIT Press, 1964.

Archer, L. Bruce.
"Systematic Method for Designers".
Design magazine, 1963-1964.

Asimow, Morris.
Introduction to Design.
Prentice-Hall, 1962.

Bagnall, Jim and Koberg, Don.
Universal Traveler.
2d ed., rev. Crisp Publications, 1990.

Banathy, Bela H.
Designing Social Systems in a Changing World.
Plenum Press, 1996.

Carmel, Erran, Whitaker, Randall D., and George, Joey F.
"PD and Joint Application Design: A Transatlantic Comparison."
Communications of the ACM, Vol. 36, No. 4, June 1993.

Cooper, Alan.
About Face: The Essentials of User Interface Design
IDG Books, 1995.

Cooper, Alan.
The Inmates Are Running the Asylum: Why High-Tech Products Drive Us Crazy and How to Restore the Sanity
SAMS, 1999.

Cross, Nigel.
Developments in Design Methodology.
John Wiley & Sons, 1984.

Cross, Nigel.
Engineering Design Methods: Strategies for Product Design.
3d ed. John Wiley & Sons, 2000.

Dubberly, Hugh.
"Alan Cooper and the Goal-directed Design Process,"
AIGA Gain, Volume 1, Number 2, 2001

Ertas, Atila , and Jones, Jesse C.
The Engineering Design Process.
2d ed., John Wiley & Sons,1996.

Goldsmith, Robin F. and Graham, Dorothy.
"The Forgotten Phase."
Software Development, July, 2002.

Hirschberg, Morton.
"The V Model."
CrossTalk, The Journal of Defense Software Engineering, June, 2000.

ISO 13407: 1999, *Human-centered design processes for interactive systems*.

Jones, John Chris.
Design Methods.
2d ed., rev. Van Nostrand Reinhold, 1992.

Kroll, Per.
"Transitioning from waterfall to iterative development."
IBM developerWorks web site, 2004.

Kruchten, Philippe.
"Going Over the Waterfall with the RUP."
IBM developerWorks web site, 2004.

Kruchten, Philippe.
"What Is thee Rational Unified Process?"
The Rational Edge e-zine, 2003.

Lawson, Brian.
How Designers Think: The Design Process Demystified.
2d ed. The University Press: Cambridge, 1991.

Maturana, Humberto R. and Varela, Francisco J.
The Tree of Knowledge: The Biological Roots of Human Understanding.
Shambhala Publications, 1998.

MacDonald, Nico.
What Is Web Design?
RotoVision, 2003.

Nassbaum, Bruce.
"The Power of Design"
Business Week, May 17, 2004.

Parshall, Steven A. and Peña, William M.
Problem Seeking: An Architectural Programming Primer.
4th ed. John Wiley & Sons, 2001.

Plamondon, Scott.
"Working smarter, not harder: An interview with Kent Beck."
IBM developerWorks web site, 2003.

PMI Standards Committee
A guide to the project management body of knowledge.
PMI, 1996.

Poyla, George.
How to Solve It: A New Aspect of Mathematical Method.
2d ed. Princeton University Press, 1988.

Rowe, Peter G.
Design Thinking.
The MIT Press, 1987.

Silver, Denise and Wood, Jane.
Joint Application Development.
2d ed. John Wiley & Sons, 1995.

Simon, Herbert.
The Sciences of the Artificial.
The MIT Press, 1994

Vredenburg, Karel.
"Building ease of use into the IBM user experience."
IBM Systems Journal, Vol. 42, No. 4, 2003.

Wells, Don.
Extreme Programming: A gentle introduction
Extremeprogramming.org web site, 2004.

Dubberly Design Office developed this book over many months. It began as a manilla folder with copies of processes. We completed the first “book” version as part of a project undertaken for Elaine Coleman and Sun’s Virtual Center for Innovation. Sun’s Noel Franus and Cindy Yepez were infinitely patient with the slow pace of work and politely pushed us to add descriptions to each model.

Greg Baker, Ryan Reposar, Audrey Crane, and Hugh Dubberly drew the models. It was Greg who patiently redrew Archer’s huge model bringing the diagram and Archer’s descriptive text together for the first time. Ryan assembled the book and handled the many changes.

We present this version for educational purposes only.
We have obtained no permissions to reproduce any of the models. Copyrights remain with their owners.

Copyright © 2004 Dubberly Design Office