How we got here, and what to do about it

Barry Hawkins
@barryhawkins

This talk is in no way affiliated with or endorsed by:

http://lookatmyhorsemyhorseisamazing.com

process

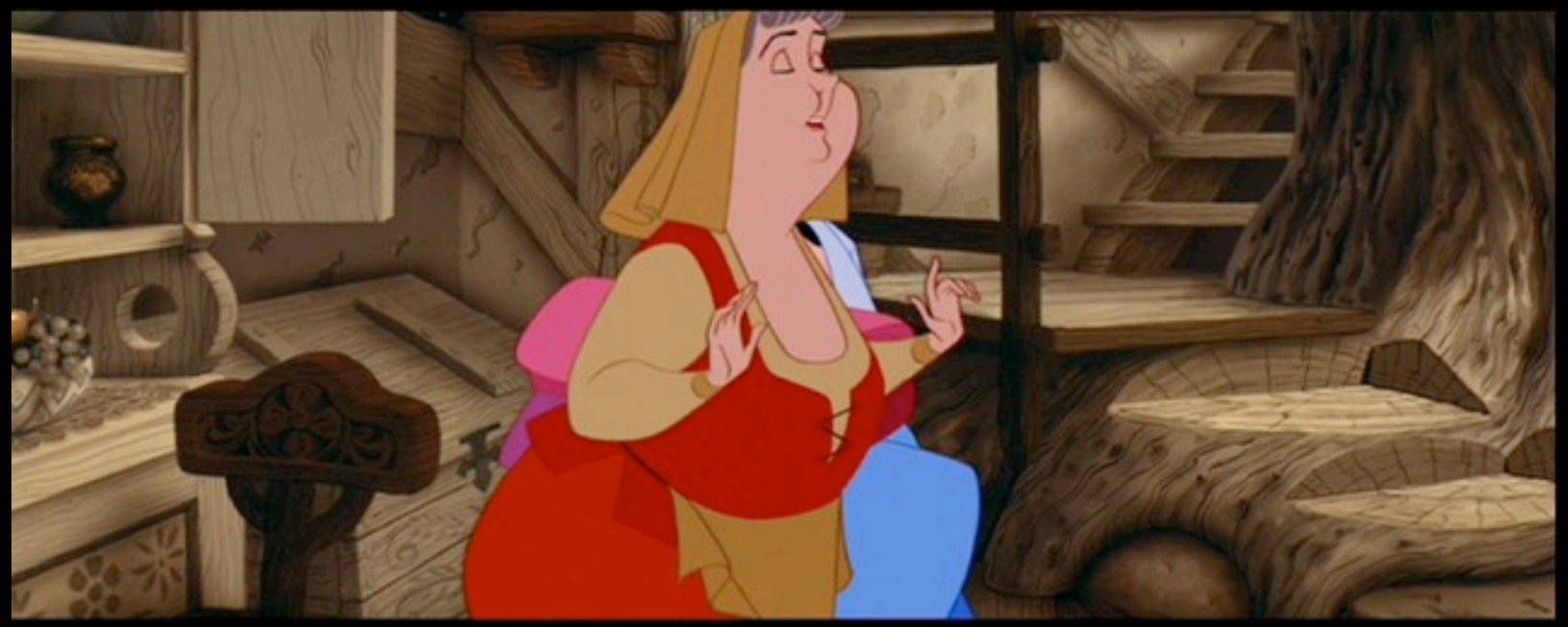
The Three Good Fairies

a real birthday cake, a dress for a princess

but I've never baked a fancy cake

I’m going to bake the cake

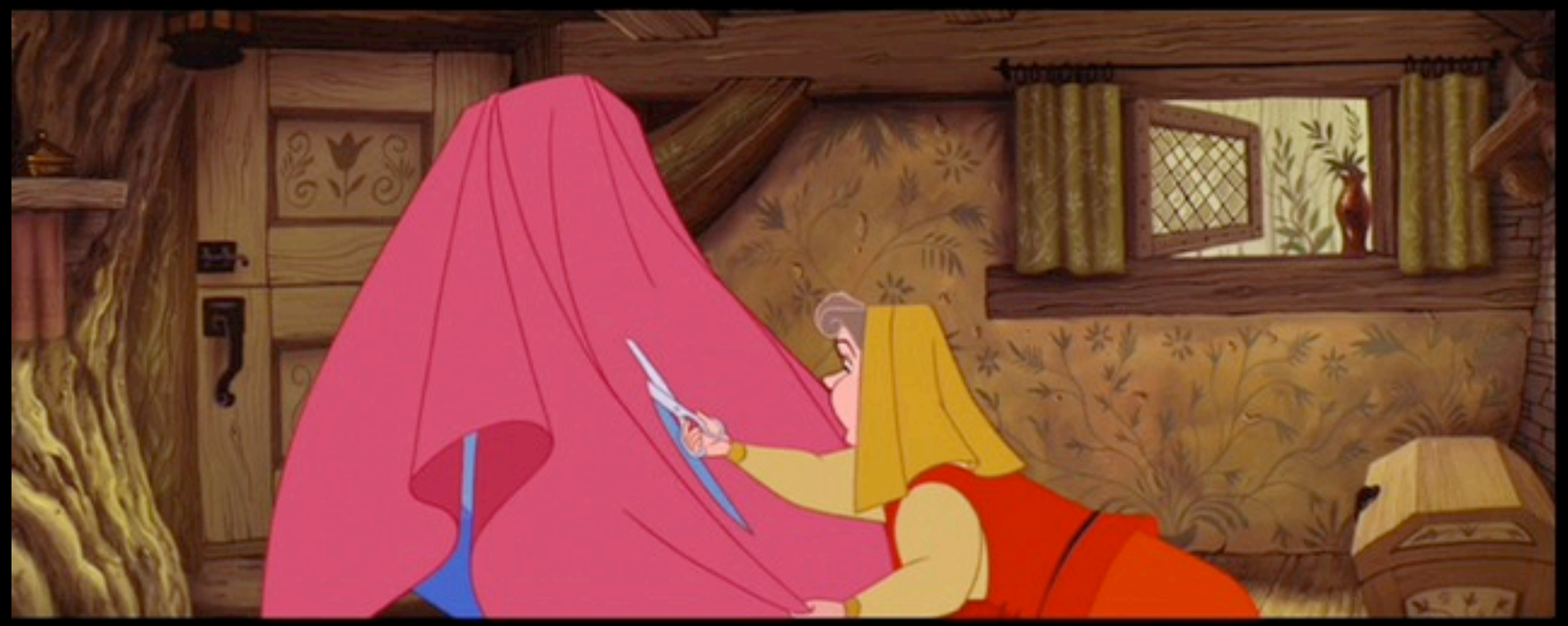
fifteen layers with pink and blue flowers



I'm making the dress

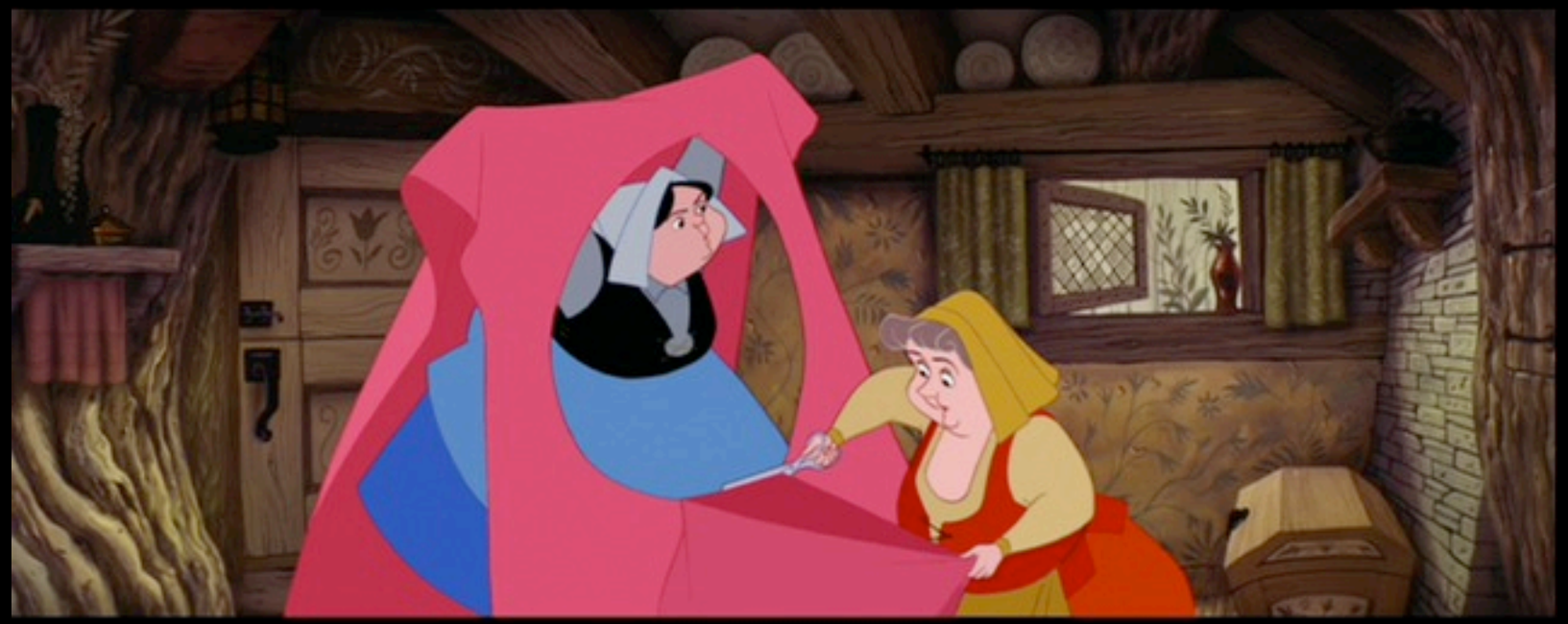
but you can't sew, and she's never cooked

it's simple, all you do is follow the book



3 cups of flour

3 cups of flour



what's that for?

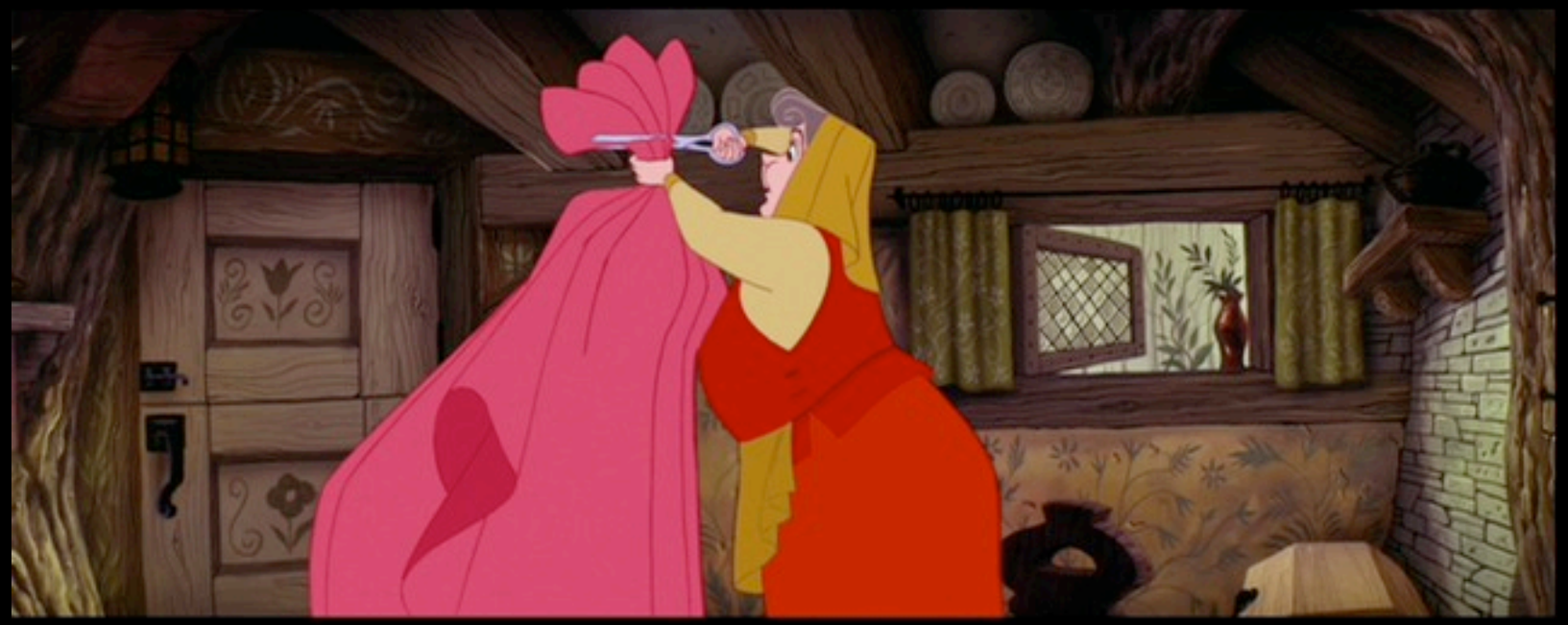
that's for the feet to go through

we decided pink

no, you decided pink

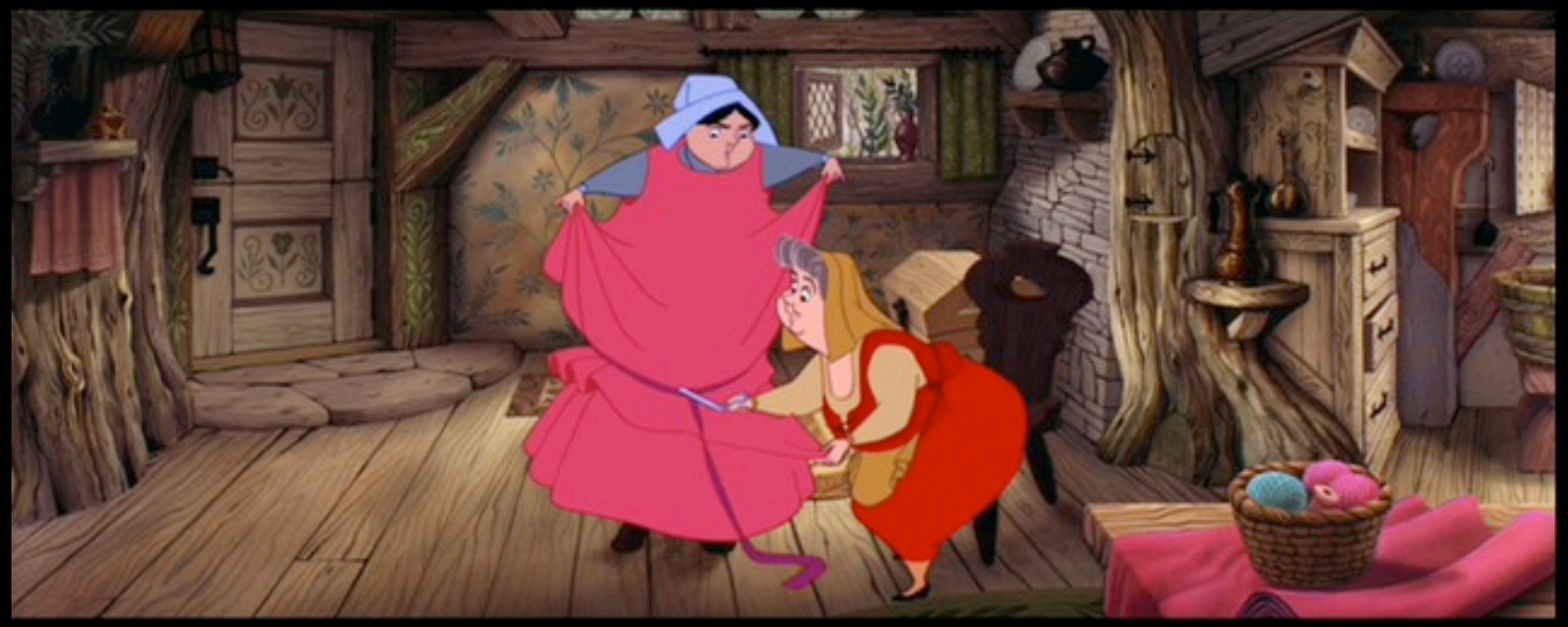
2 eggs, fold in gently

2 eggs, fold in gently



I can’t breathe

it looks awful...



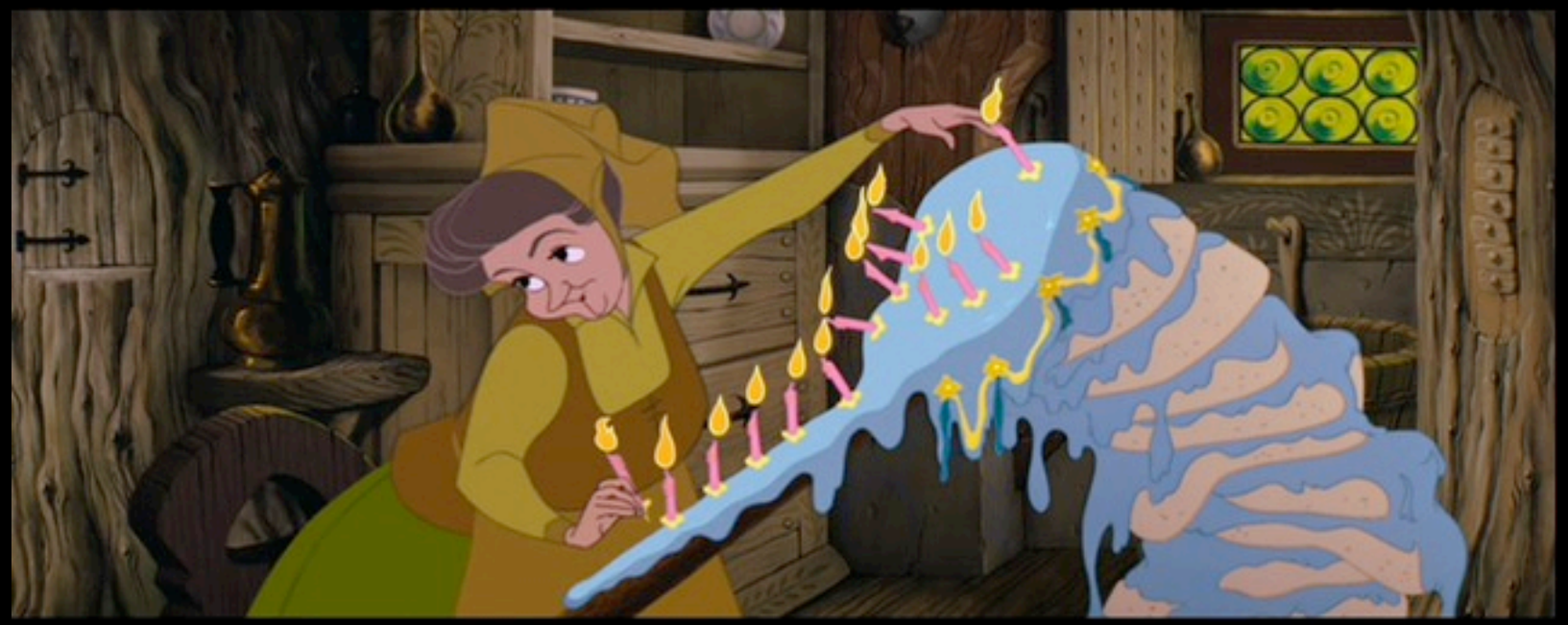
that's because it's on you, dear

there

oops

well, what do you think of it?

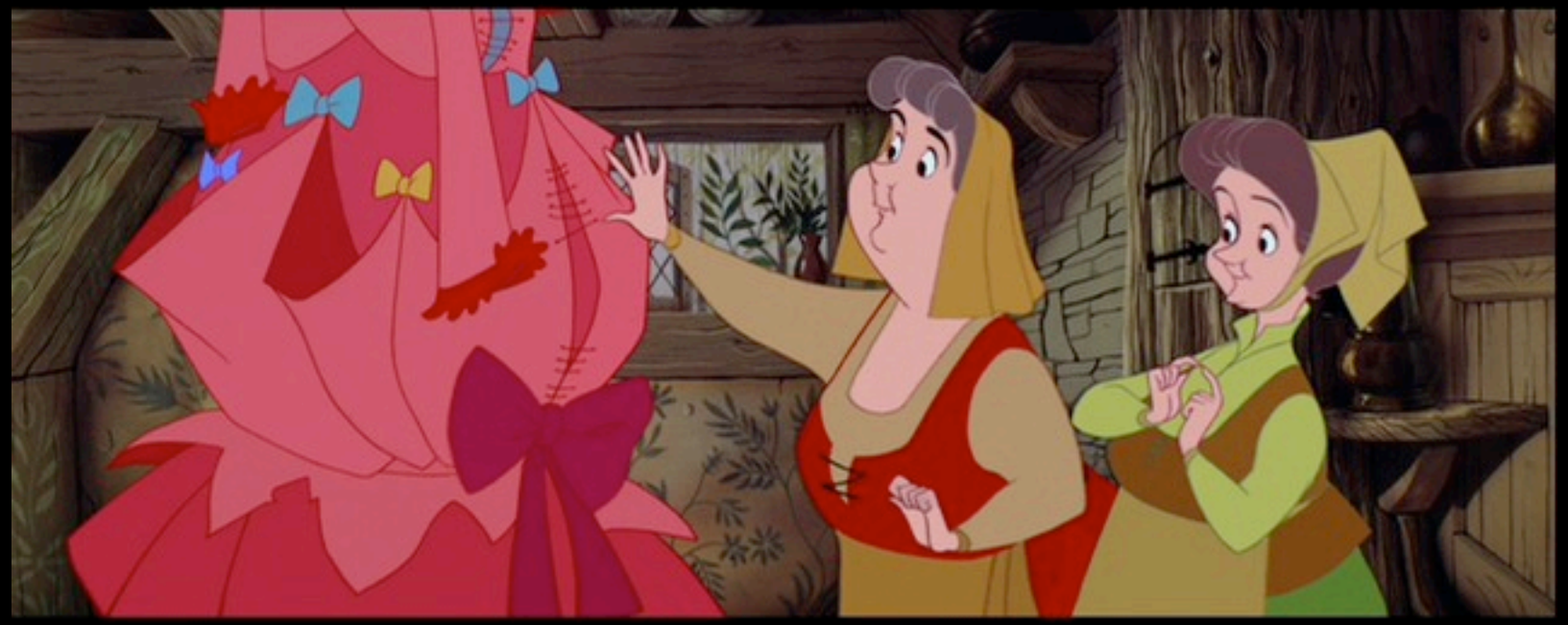
it’s a very unusual cake



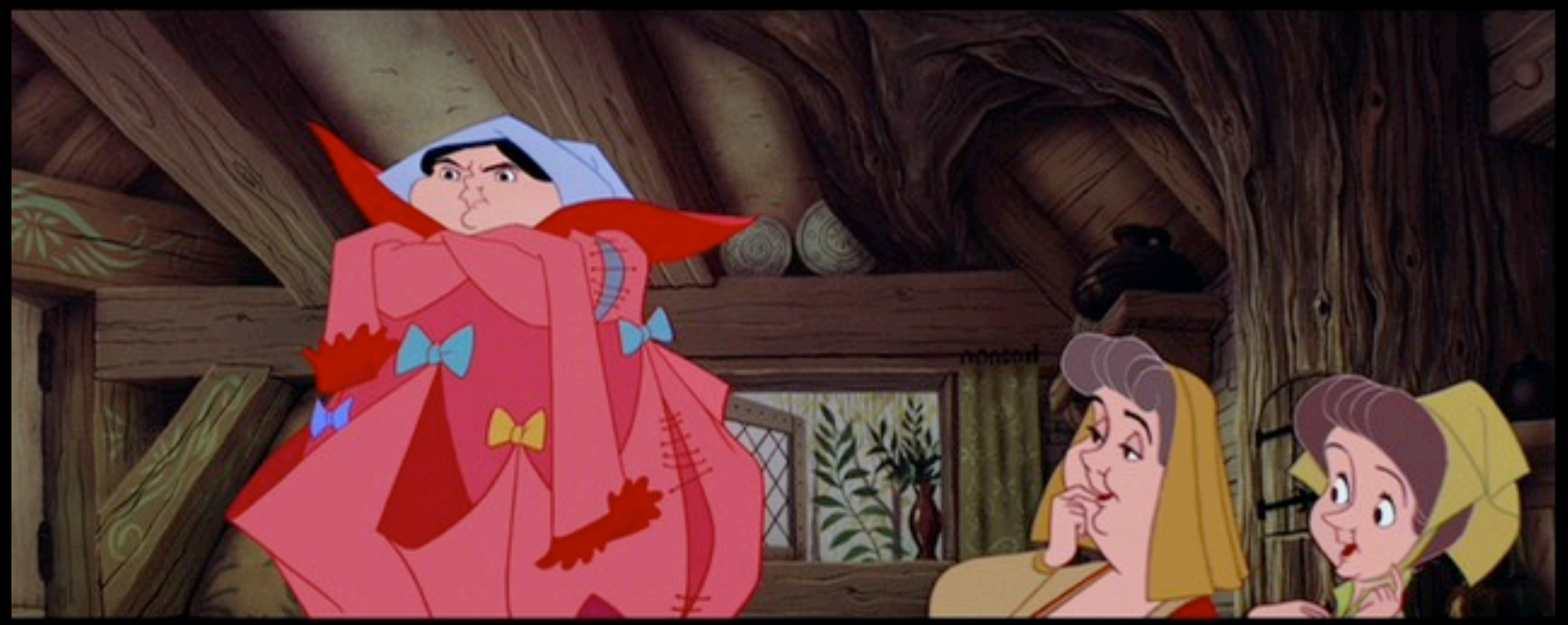
it’ll be much thicker
after it’s baked

what do you think of the dress?

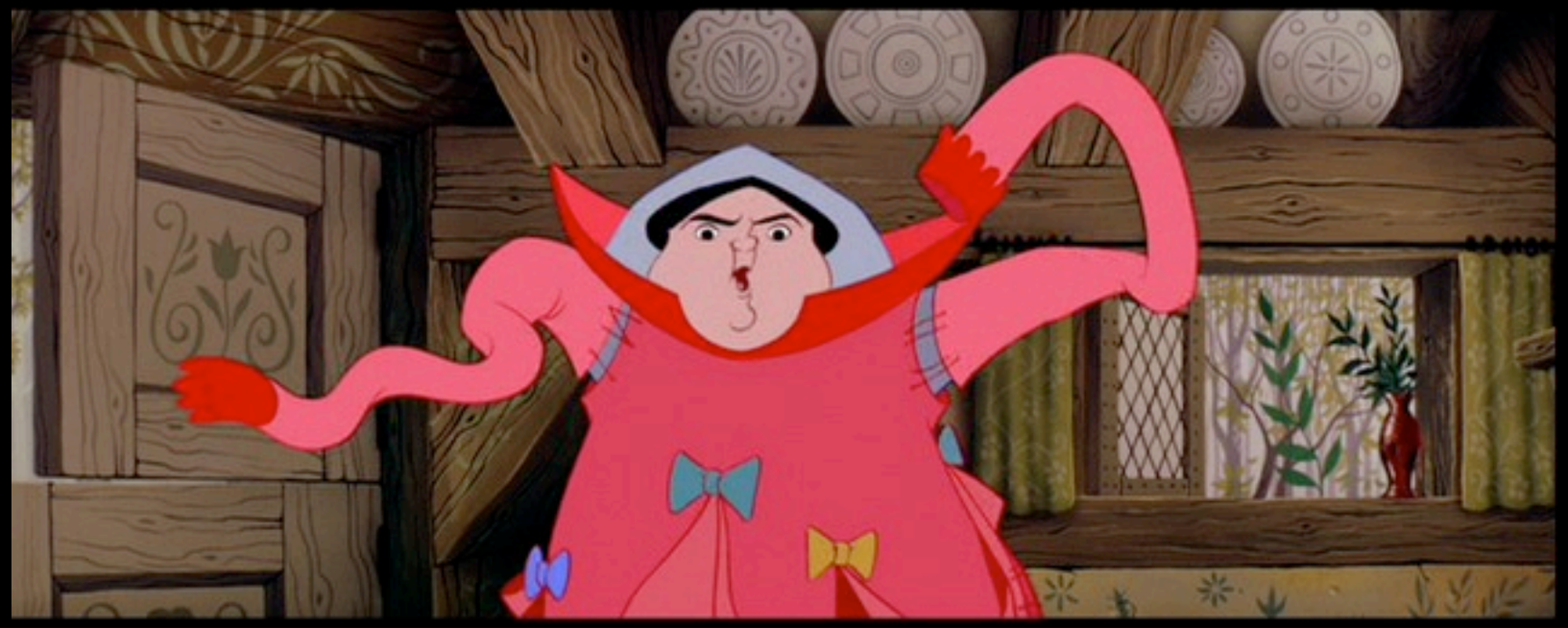
it’s not exactly the way
it is in the book



perhaps if I added a few more ruffles

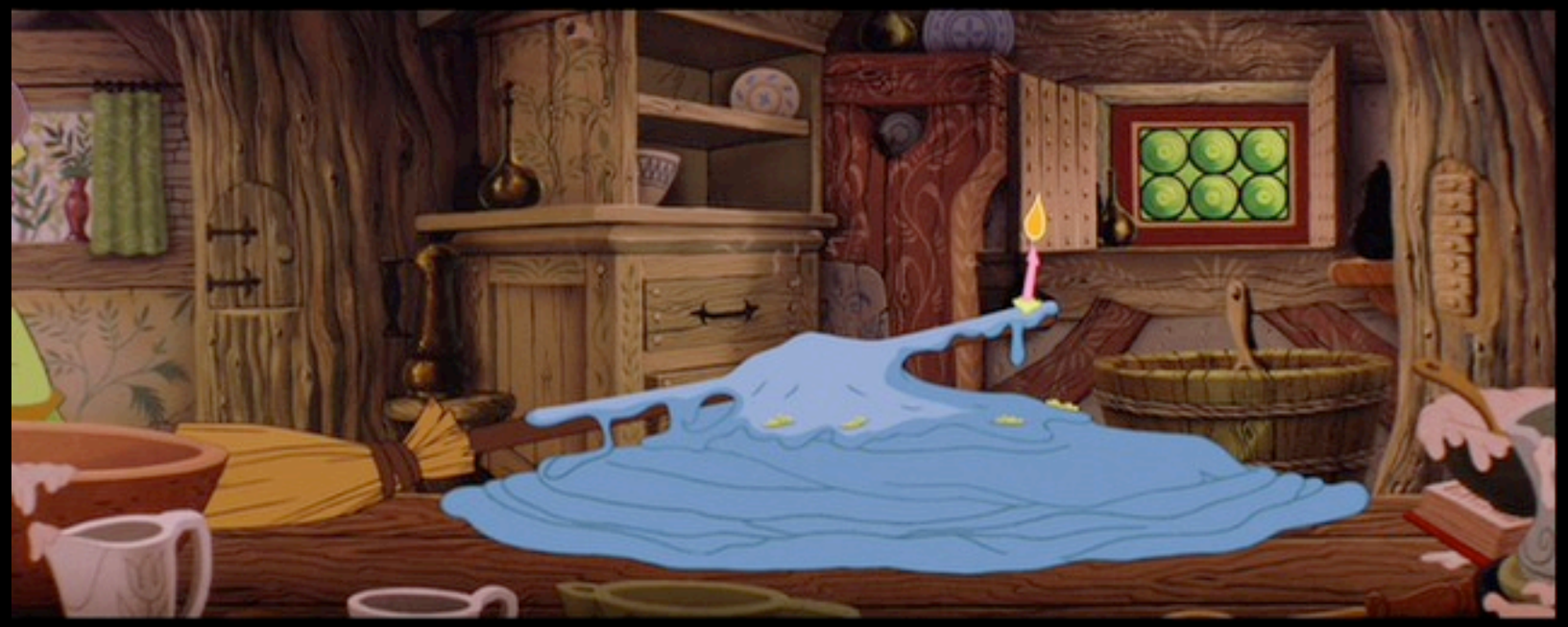


what do you think?



we have had enough of this nonsense

I think we should think of Rose



there are no magic wands

how we got here

waterfall, or something

MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS

Dr. Winston W. Royce

INTRODUCTION

I am going to describe my personal views about managing large software developments. I have had various assignments during the past nine years, mostly concerned with the development of software packages for spacecraft mission planning, commanding and post-flight analysis. In these assignments I have experienced different degrees of success with respect to arriving at an operational state, on-time, and within costs. I have become prejudiced by my experiences and I am going to relate some of these prejudices in this presentation.

COMPUTER PROGRAM DEVELOPMENT FUNCTIONS

There are two essential steps common to all computer program developments, regardless of size or complexity. There is first an analysis step, followed second by a coding step as depicted in Figure 1. This sort of very simple implementation concept is in fact all that is required if the effort is sufficiently small and if the

the “waterfall paper”

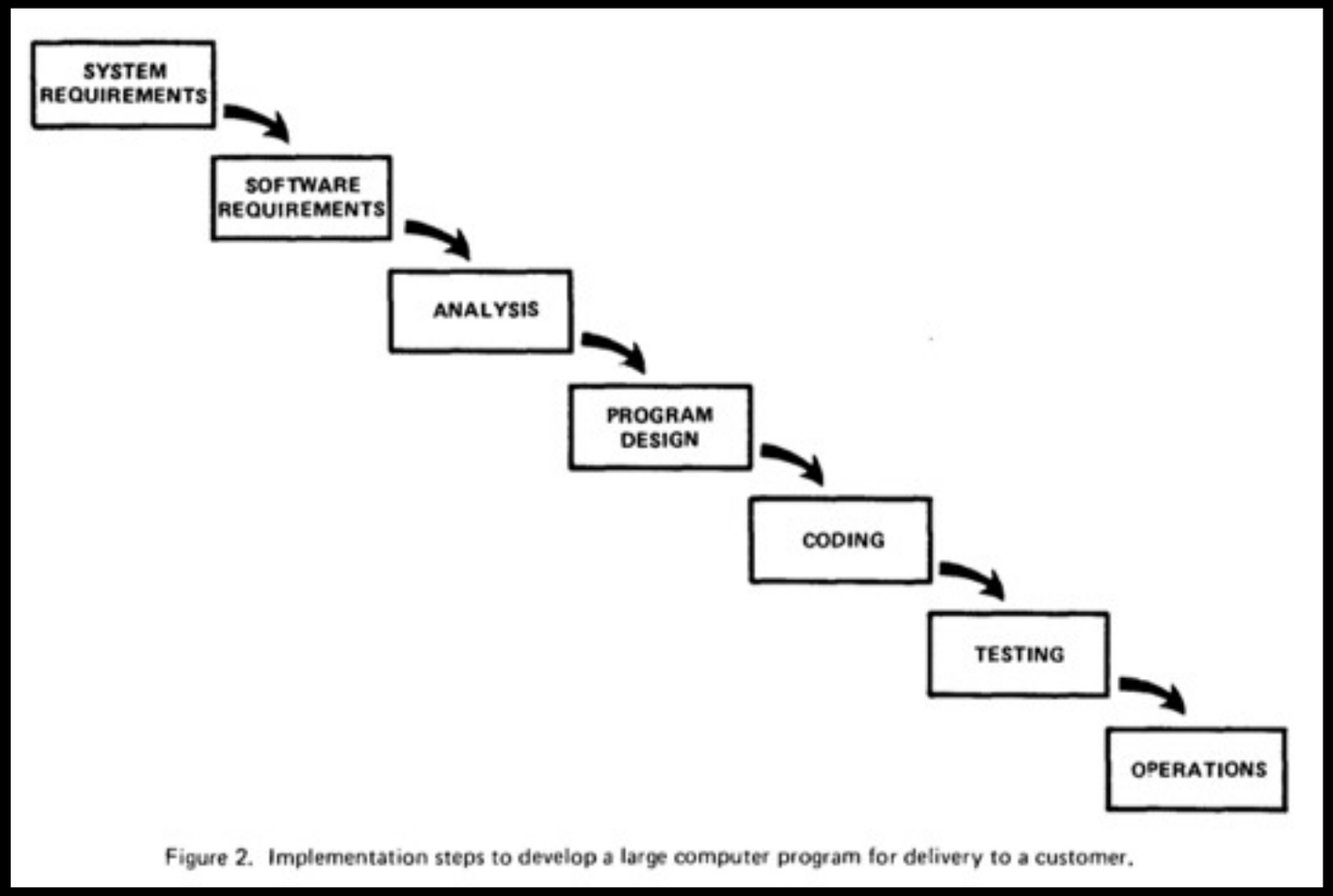


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

development steps

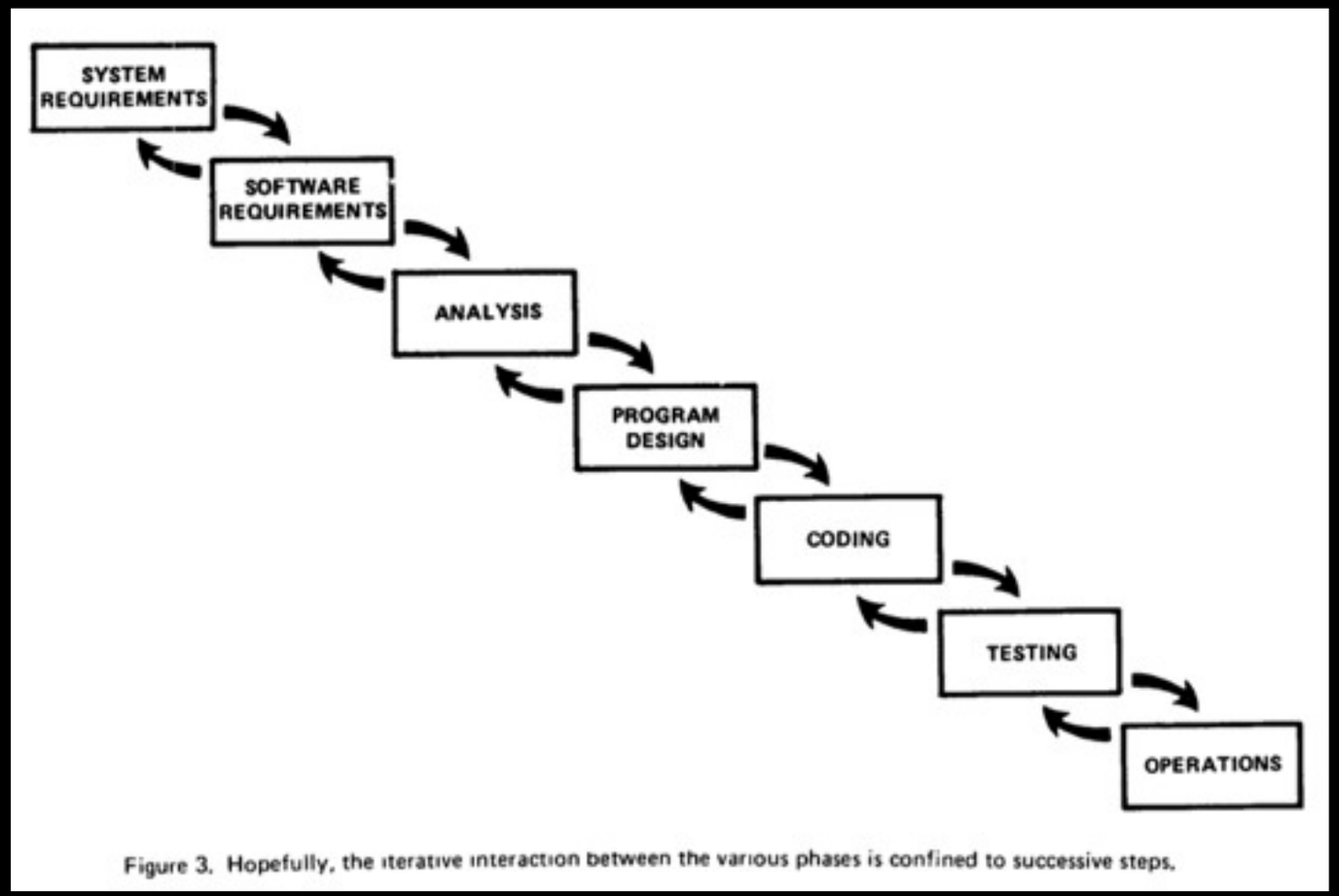


Figure 3. Hopefully, the iterative interaction between the various phases is confined to successive steps.

steps that iterate

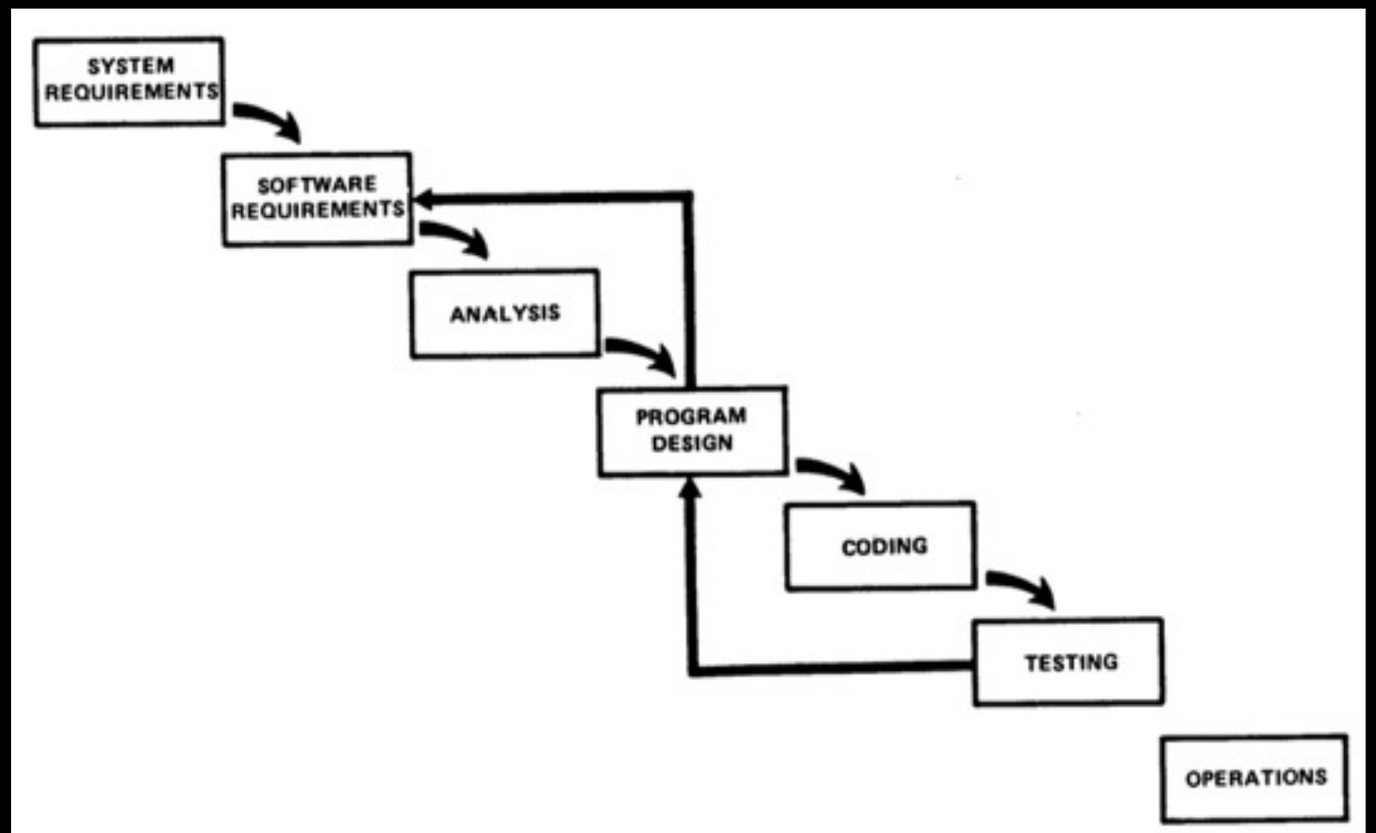


Figure 4. Unfortunately, for the process illustrated, the design iterations are never confined to the successive steps.

change hoses steps

do it twice

involve the customer

“For some reason what a software design is going to do is subject to wide interpretation even after previous agreement. It is important to involve the customer in a formal way so that he has committed himself at earlier points before final delivery. To give the contractor free rein between requirement definition and operation is inviting trouble.” - Dr. Winston W. Royce

Semi-Automatic Ground Environment (SAGE)

Production of Large Computer Programs

HERBERT D. BENINGTON

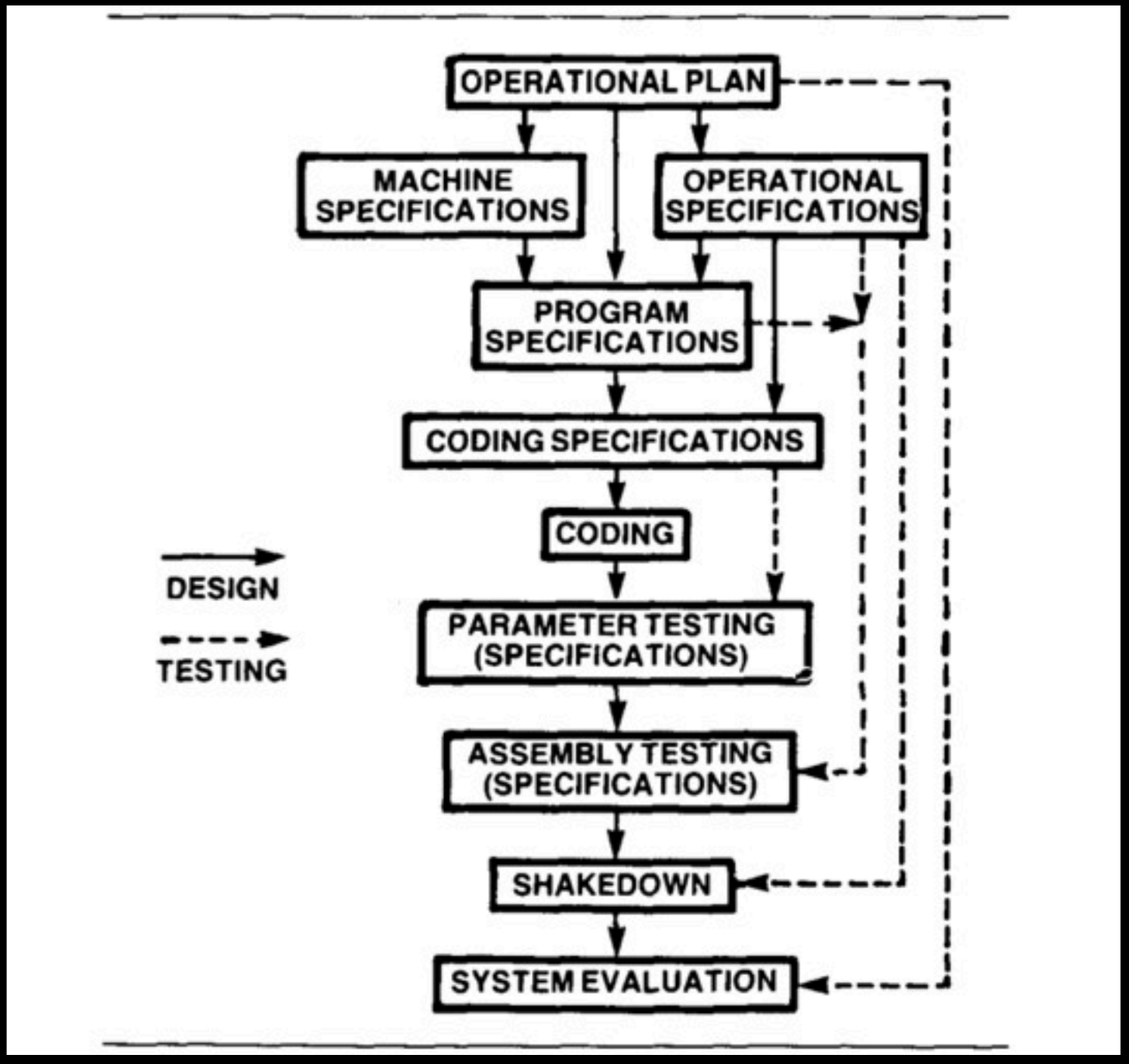
The paper is adapted from a presentation at a symposium on advanced programming methods for digital computers sponsored by the Navy Mathematical Computing Advisory Panel and the Office of Naval Research in June 1956. The author describes the techniques used to produce the programs for the Semi-Automatic Ground Environment (SAGE) system.

Categories and Subject Descriptors: K. 2 [History of Computing]—SAGE, software, systems
General Terms: Design, Management
Additional Key Words and Phrases: Lincoln Laboratory

Editor's Note

When we all began to work on SAGE, we believed our own myths about software—that one can do anything with software on a general-purpose computer; that software is easy to write, test, and maintain; that it is easily replicated, doesn't wear out, and is not subject to transient errors. We had a lot to learn.

As Herb Benington discusses in the following paper, we had already successfully written quite a lot of software for experimental purposes. We were misled by the success we had had with capable engineers writing programs that were small enough for an individual to understand fully. With SAGE, we were faced with programs that were too large for one person to grasp entirely and also with the need to hire and train large numbers of people to become programmers—after all, there were only a handful of trained programmers in the whole world. We were faced with organizing and managing a whole new art


OPERATIONAL PLAN
MACHINE SPECIFICATIONS
OPERATIONAL SPECIFICATIONS
PROGRAM SPECIFICATIONS
CODING SPECIFICATIONS
CODING
PARAMETER TESTING (SPECIFICATIONS)
ASSEMBLY TESTING (SPECIFICATIONS)
SHAKEDOWN
SYSTEM EVALUATION
DESIGN
TESTING

accept that testing will be sampling only

initial testing should use simulated inputs

top-down programming

"The great majority seemed to espouse the following approach: we must write the initial top-down specification (for example, the A Spec), then the next one (typically, the B Spec), so we will know precisely what our objectives are before we produce one line of code. This attitude can be terribly misleading and dangerous. To stretch an analogy slightly, it is like saying that we must specify the characteristics of a rocket engine before measuring the burning properties of liquid hydrogen." - Herbert D. Bennington

rup

the six best practices

iterate with risk as driver

manage requirements

employ component-based architecture

model software visually

continuously verify quality

control changes

tools, glorious tools

scrum and xp

the agile manifesto

individuals and interactions over processes and tools

working software over comprehensive documentation

customer collaboration
over contract
negotiation

responding to change
over following a plan

lean

standardized work

value stream mapping

visual control

what to do about it

few new ideas, mostly new expressions

pursue understanding

don’t make process a religion

realize the primacy of culture

commoditization of

excellence is a myth

"Lean programs often tend to gloss over this detail and instead become more enamored with the more abstract improvement concepts of "flow" or "pull" for example. The second point of concern was the tendency in many programs to elevate mere tools (standardized work, value stream mapping, visual control, etc.) to an unhealthy status beyond their design intent." - Art Smalley

embrace hard work,

dispel myths of ease

“It's great that you can assemble a car in one minute and eliminate waste. For the company, it's an economically efficient way of making cars. But I understand Europeans get breaks. We, too, should have this humane touch in our system. You should at least have a second to wipe the sweat off your brow.” - Kunioshi Ishida, Toyota Factory Worker

"I've always been impressed by the fact that workers talk about it like it's a normal thing, but it's not. For example, when walking down the corridor from one office to another, you're supposed to turn at right angles. You're not allowed to cut corners." - Shunichi Sakae, Retired Toyota Factory Worker

internal adoption levels vary

"But there are other ways of applying the Toyota Way. Some Toyota designers and engineers treat the workers as disposable, just like a machine. They give them big burdens and try to extract the maximum from them." - Shigenobu Matsubara, Toyota Assembly Line Designer

use consultants

sparingly

simply begin

"...[P]roducing large computer programs is like raising a family. You can observe your neighbors and see all of the successes and failures in their children. You can reflect on the experiences you had as one member of a large family. You can observe all the proper maxims of life and society. You can even study at length the experiences of many others who have raised families. In the final analysis, however, you have to start out and do it on your own, learn the unique options you have, see what unexpected problems arise, and, with reasonable luck, perform about as well as those who have been doing it forever." - Herbert Benington, 1983

http://j.mp/GDSjFD