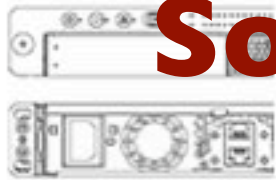


Apple and the Global Environment

At Apple, environmental considerations are an integral part of our business practices. From the earliest stages of design through manufacturing, use and recycling, we take care to minimize the environmental impact of our activities.

Lifecycle phases and their environmental considerations



Product Design

Selecting environmentally acceptable materials and substances. Developing features to optimize the energy efficiency of our systems and to facilitate recycling.



Manufacturing

Reducing waste — water, emissions, by-products — from the manufacturing process. Continually improving the energy efficiency of our manufacturing facilities.



Use

Providing software tools that enable users to control the energy-saving features of their systems.



Recycling/EOL

Providing product recycling programs for our customers worldwide. Diverting a high percentage of returned equipment from landfill.

Three areas in which we focus considerable attention are materials management, energy efficiency and recycling, which are key elements of a product's lifecycle. Our efforts in these areas have resulted in some impressive environmental successes.

Learn More

At Apple, our commitment to the environment is a core value. To read more about the steps we take to foster environmental improvements in our products and the way we do business, visit our environmental [website](#).



will learn about these important achievements and more.

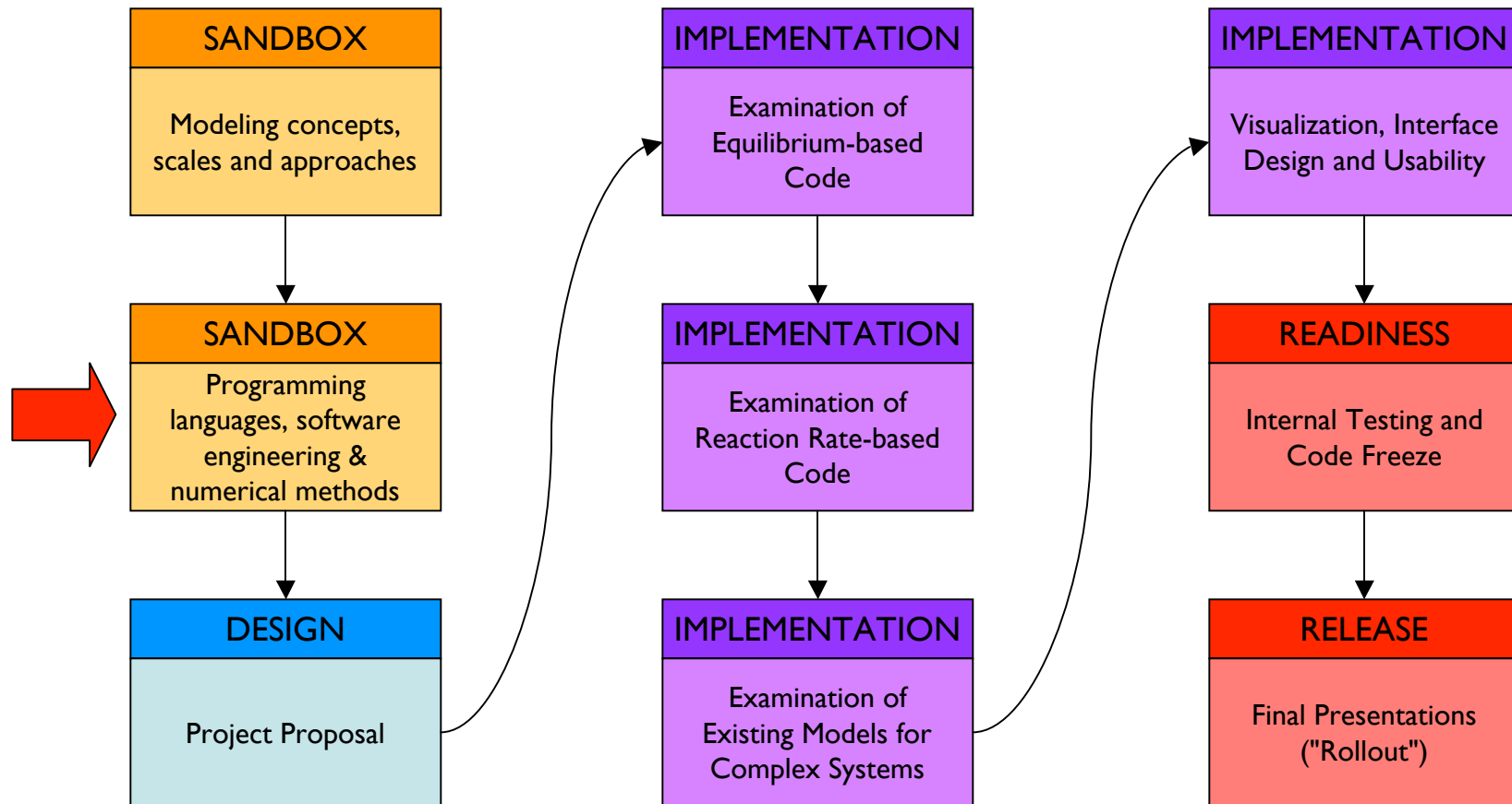
- Between the first-generation and current-generation iMac, improvements in CPU power management enabled an 87% decrease in sleep-mode power usage.
- Our Customer-Installable Parts program puts the power of system expandability in the consumer's hands, helping to extend the useful life of a computer.



Programming Languages, Software Engineering and Numerical Methods

CEE 373

Roadmap

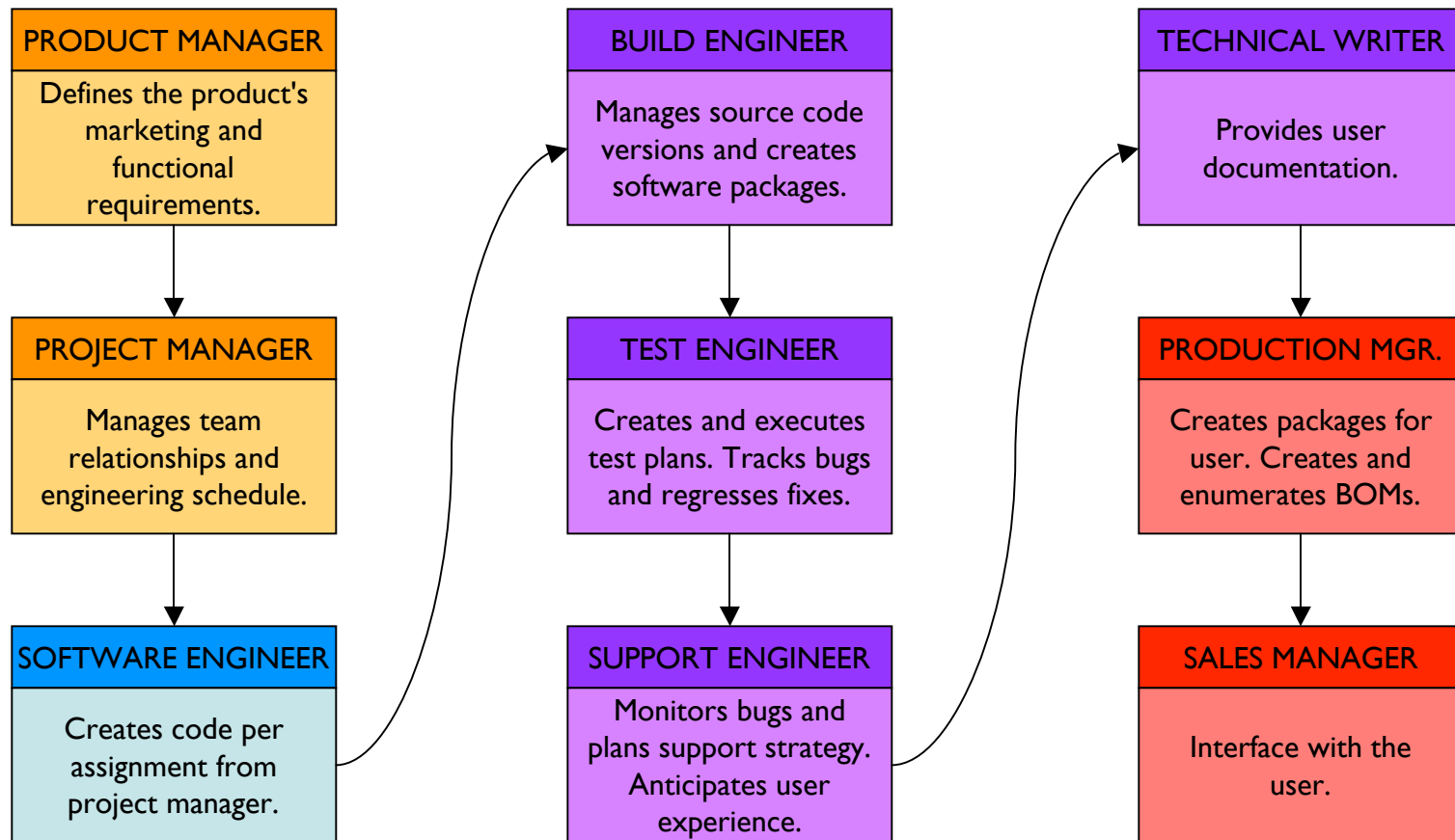


Computing in a Nutshell

Operating Systems	Programming	Platforms
Linux Windows Unix MacOS PalmOS Symbian	Assembly C/C++/C# Java/J++/J# Fortran BASIC PHP	Desktop Laptop Handheld Server Embedded Networked/Parallel

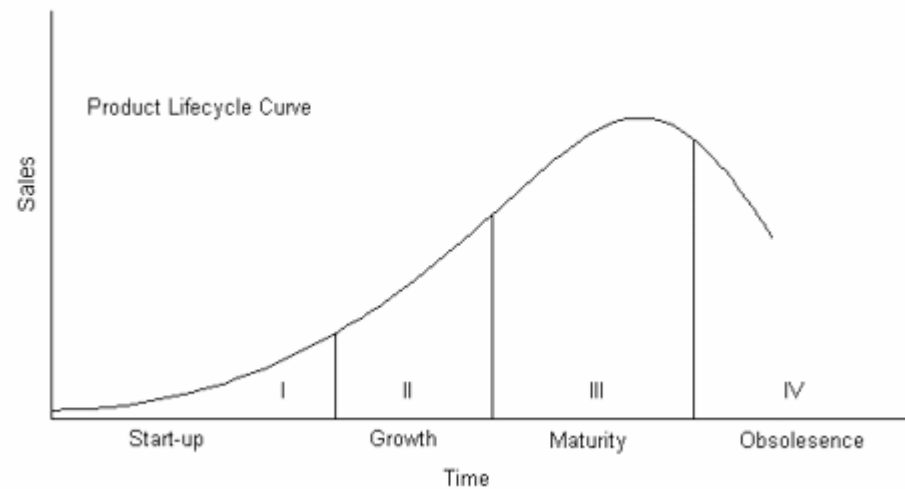


The Cross-Functional Team



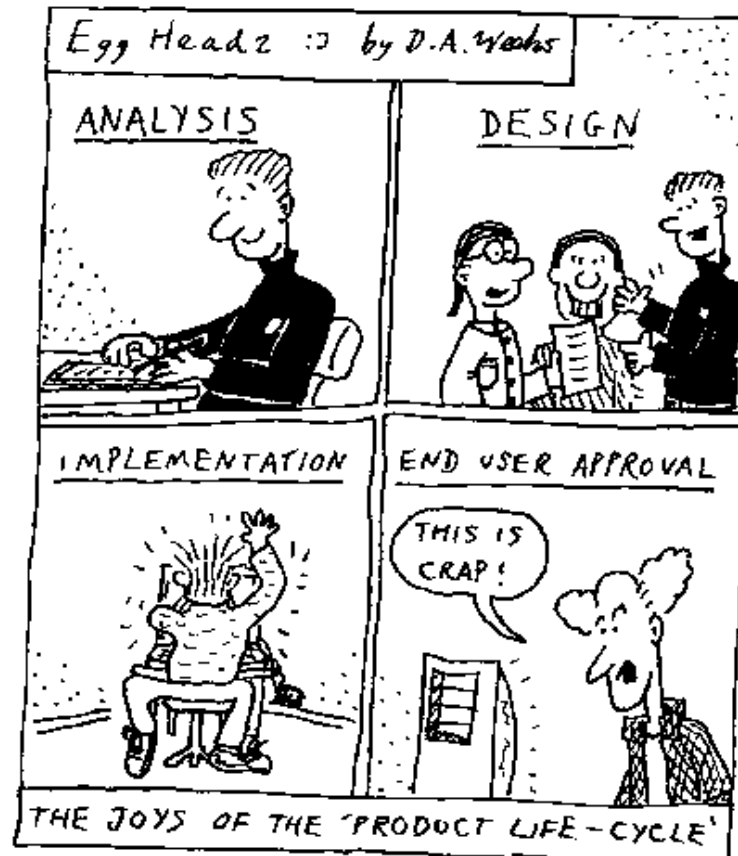
Software Engineering

- Product Life Cycles

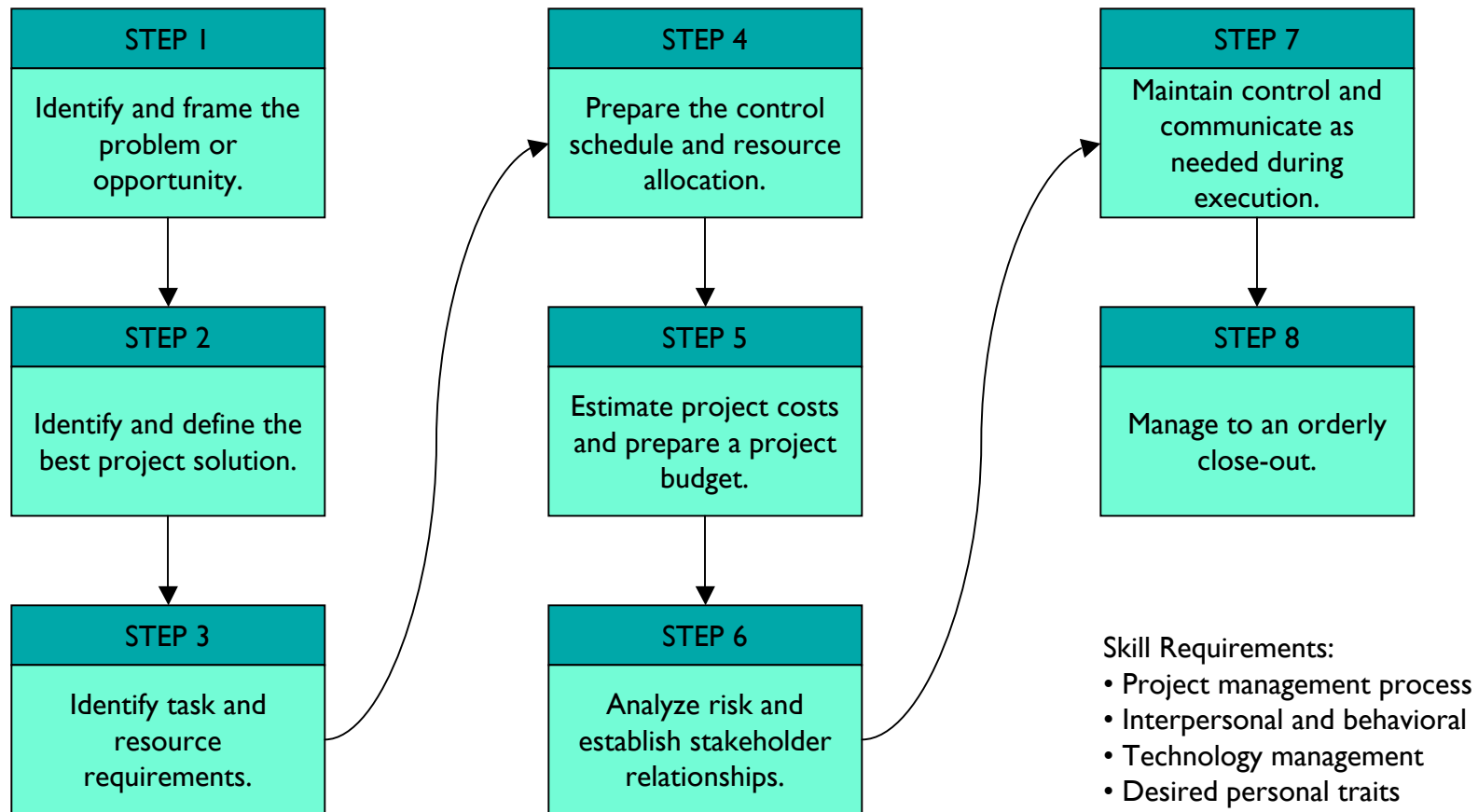


Software Engineering

- Product Life Cycles

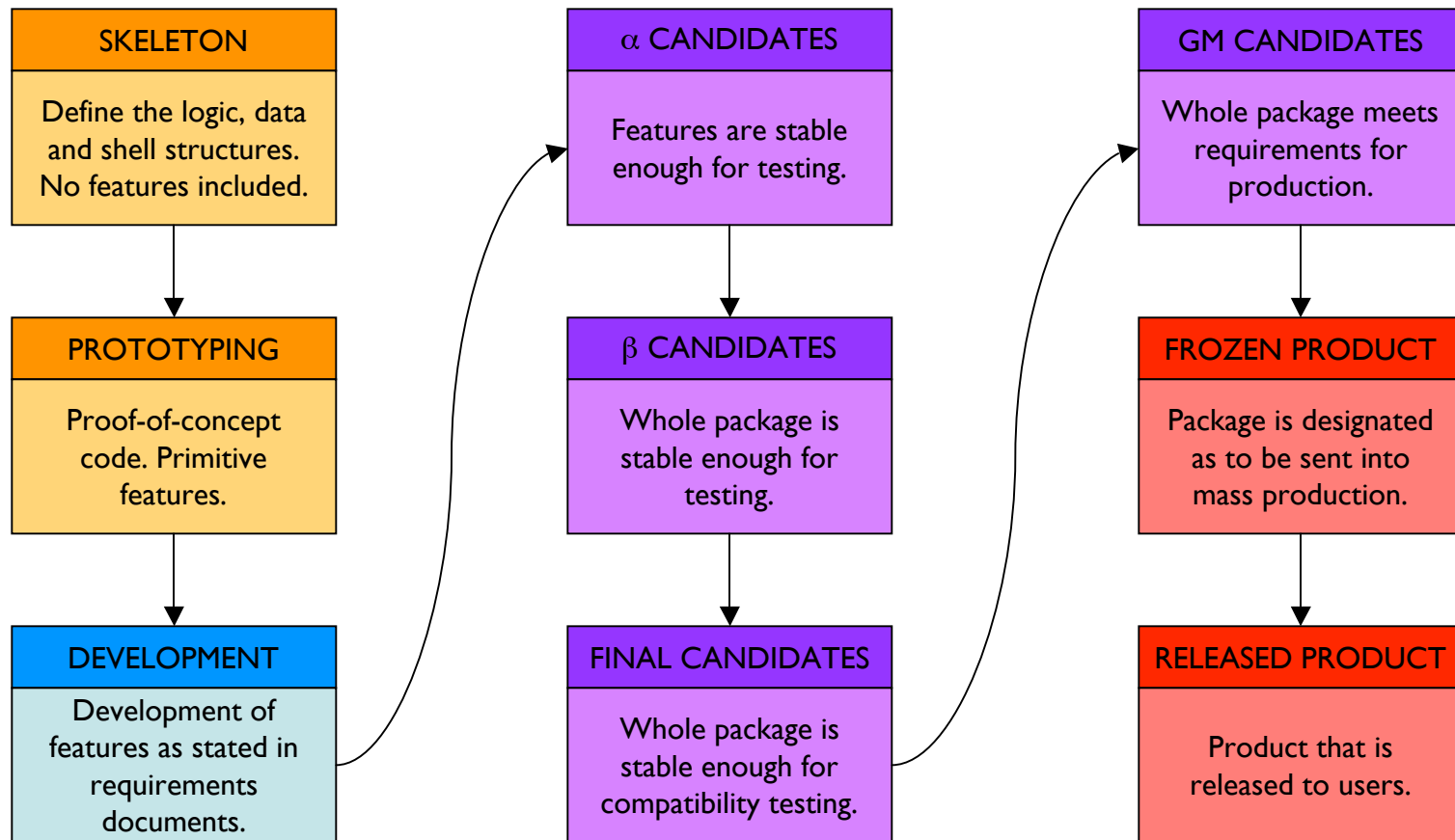


Project Management Process

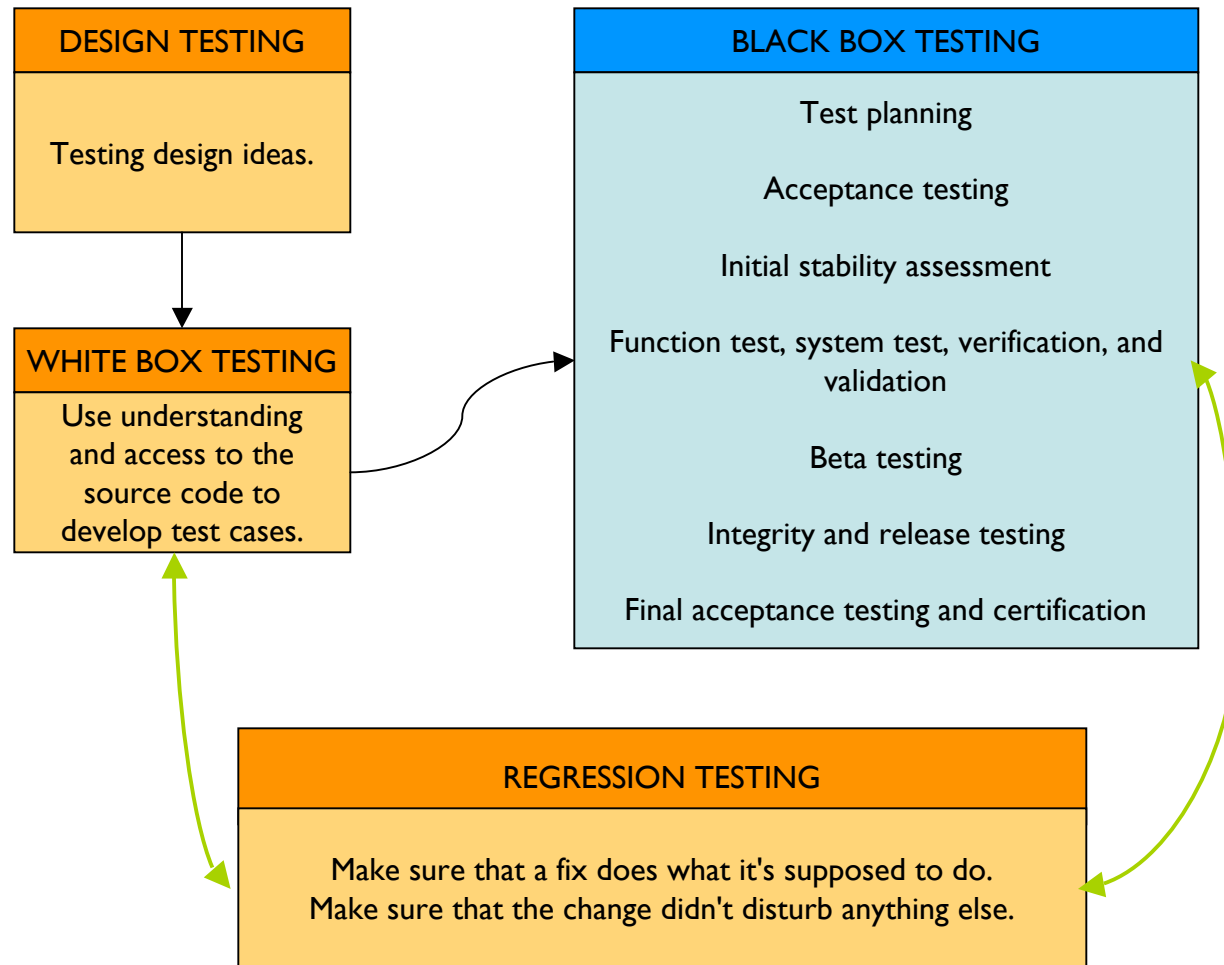


After Heerkens, 2002

Software Engineering



Software Quality Assurance



Bugs



REPORTING & ANALYZING	TRACKING
TESTER Explain how to reproduce the problem. Analyze the error so you can describe it in a minimum number of steps. Write a report that is complete, easy to understand, and non-antagonistic. Assigns severity rating. PROJECT MANAGER Assigns bug to a programmer. Assigns priority rating and current status.	Team members can track the status of a bug. Project Manager produces periodic bug reports. Project Manager leads bug review meetings to decide whether bugs are fixed, not fixed, causing a new problem, irreproducible, deferred, or not a bug. Deferrals are used to "close" all lingering bugs so a product can ship to a user. Tracking system metrics is used to compare growing number of bugs versus rate of closure.

Numerical Methods

15.2 Fitting Data to a Straight Line

A concrete example will make the considerations of the previous section more meaningful. We consider the problem of fitting a set of N data points (x_i, y_i) to a straight-line model

$$y(x) = y(x; a, b) = a + bx \quad (15.2.1)$$

$$\begin{aligned} \Delta &\equiv SS_{xx} - (S_x)^2 \\ a &= \frac{S_{xx}S_y - S_xS_{xy}}{\Delta} \\ b &= \frac{SS_{xy} - S_xS_y}{\Delta} \end{aligned} \quad (15.2.6)$$

Equation (15.2.6) gives the solution for the best-fit model parameters a and b .

Press et al, 1992

Numerical Methods

`void fit(float x[], float y[], int ndata, float sig[], int mwt, float *a, float *b, float *siga, float *sigb, float *chi2, float *q)`
Given a set of data points `x[1..ndata]`, `y[1..ndata]` with individual standard deviations `sig[1..ndata]`, fit them to a straight line $y = a + bx$ by minimizing χ^2 . Returned are `a`, `b` and their respective probable uncertainties `siga` and `sigb`, the chi-square `chi2`, and the goodness-of-fit probability `q` (that the fit would have χ^2 this large or larger). If `mwt=0` on input, then the standard deviations are assumed to be unavailable: `q` is returned as 1.0 and the normalization of `chi2` is to unit standard deviation on all points.

```
{
    float gammq(float a, float x);
    int i;
    float wt,t,sxoss,sx=0.0,sy=0.0,st2=0.0,ss,sigdat;

    *b=0.0;
    if (mwt) {
        ss=0.0;
        for (i=1;i<=ndata;i++) {
            wt=1.0/SQR(sig[i]);
            ss += wt;
            sx += x[i]*wt;
            sy += y[i]*wt;
        }
        // Accumulate sums ...
        // ...with weights
    } else {
        for (i=1;i<=ndata;i++) {
            sx += x[i];
            sy += y[i];
        }
        // ...or without weights.
        ss=ndata;
    }
}
```

Press et al, 1992

Numerical Methods

```

sxoss=sx/ss;
if (mwt) {
    for (i=1;i<=ndata;i++) {
        t=(x[i]-sxoss)/sig[i];
        st2 += t*t;
        *b += t*y[i]/sig[i];
    }
} else {
    for (i=1;i<=ndata;i++) {
        t=x[i]-sxoss;
        st2 += t*t;
        *b += t*y[i];
    }
}
*b /= st2;
*a=(sy-sx*(*b))/ss;
*sig=sqrt((1.0+sx*sx/(ss*st2))/ss);
*sigb=sqrt(1.0/st2);

```

Solve for a , b , σ_a , and σ_b .

Press et al, 1992

Numerical Methods

9.4 Newton-Raphson Method Using Derivative

Perhaps the most celebrated of all one-dimensional root-finding routines is *Newton's method*, also called the *Newton-Raphson method*. This method is distinguished from the methods of previous sections by the fact that it requires the evaluation of both the function $f(x)$, and the derivative $f'(x)$, at arbitrary points x . The Newton-Raphson formula consists geometrically of extending the tangent line at a current point x_i until it crosses zero, then setting the next guess x_{i+1} to the abscissa of that zero-crossing (see Figure 9.4.1). Algebraically, the method derives from the familiar Taylor series expansion of a function in the neighborhood of a point,

$$f(x + \delta) \approx f(x) + f'(x)\delta + \frac{f''(x)}{2}\delta^2 + \dots \quad (9.4.1)$$

For small enough values of δ , and for well-behaved functions, the terms beyond linear are unimportant, hence $f(x + \delta) = 0$ implies

$$\delta = -\frac{f(x)}{f'(x)}. \quad (9.4.2)$$

Numerical Methods

```

rts=0.5*(x1+x2);
dxold=fabs(x2-x1);
dx=dxold;
(*funcd)(rts,&f,&df);
for (j=1;j<=MAXIT;j++) {
    if (((rts-xh)*df-f)*((rts-xl)*df-f) > 0.0)
        || (fabs(2.0*f) > fabs(dxold*df)) {
        dxold=dx;
        dx=0.5*(xh-xl);
        rts=xl+dx;
        if (xl == rts) return rts;
    } else {
        dxold=dx;
        dx=f/df;
        temp=rts;
        rts -= dx;
        if (temp == rts) return rts;
    }
    if (fabs(dx) < xacc) return rts;
    (*funcd)(rts,&f,&df);
}

```

Initialize the guess for root,
the "stepsize before last,"
and the last step.

Loop over allowed iterations.
Bisect if Newton out of range,
or not decreasing fast enough.

Change in root is negligible.
Newton step acceptable. Take it.

Convergence criterion.

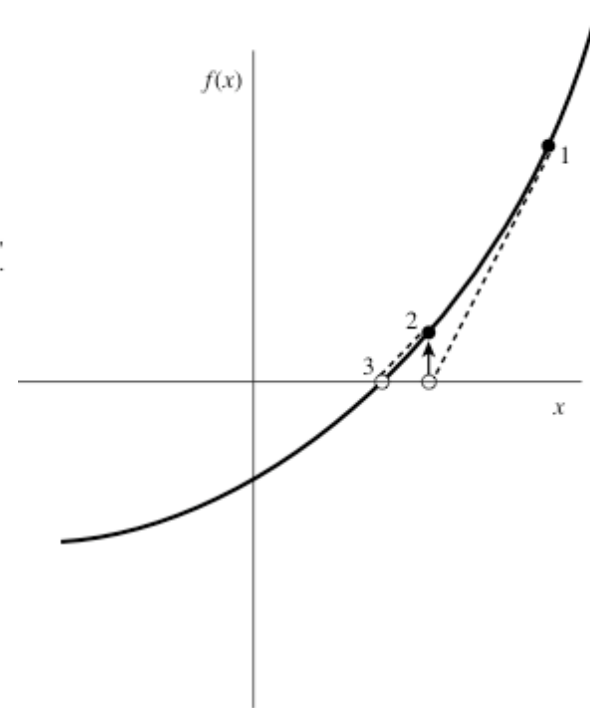
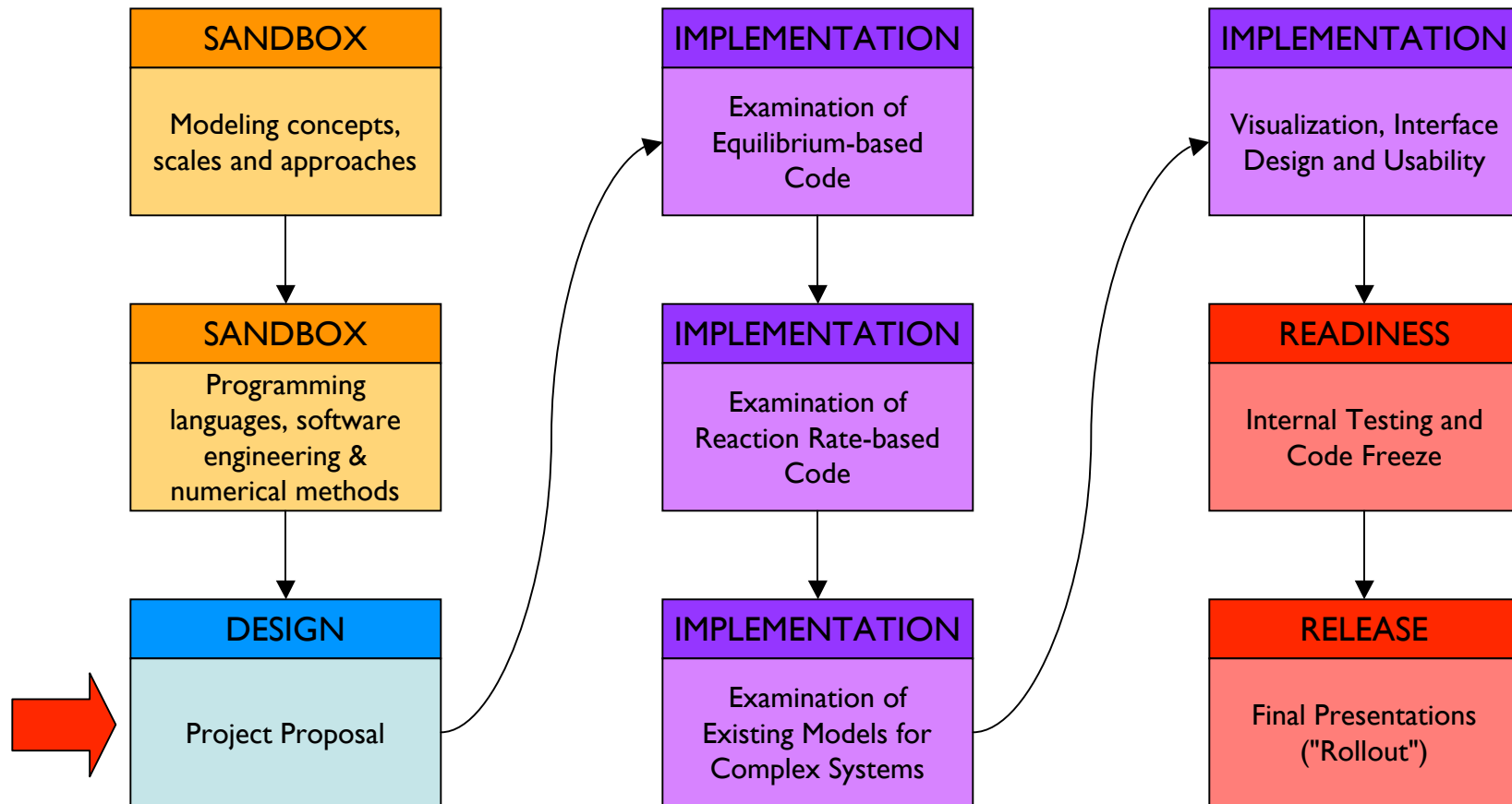


Figure 9.4.1. Newton's method extrapolates the local derivative to find the next estimate of the root. In this example it works well and converges quadratically.

Press et al, 1992

Roadmap



Assignment: Project Proposal

A. SCIENTIFIC LITERATURE REVIEW

Provide background information on the scientific or engineering area with which the project is related.

B. CONCEPTUAL DESIGN

Description of compartments and processes. Schematic of model environment.

C. MARKETING REQUIREMENTS

Document how there is a need for the proposed product. Show how there is differentiation from existing products. Use information at <http://www.epri.com/eprisoftware/processguide/srdc.html> as a guideline.

D. FUNCTIONAL REQUIREMENTS

Provide a description of the software including functional capabilities and user interactions. Include how the user will learn to use your product. Check out <http://www.epri.com/eprisoftware/processguide/fscontents.html> as a guideline.

E. TEST PLAN

Create a list of tests that can be applied to the product to make sure that it is functional and usable, and meets the documented requirements.

F. PROJECT SCHEDULE

Create a timeline showing project tasks, resource allocation, durations and relationships.. Be sure to declare the frequency of checking the schedule during the project to determine if adjustments to the schedule need to be made.