How big companies use systems architecting

Neil Siegel, Ph.D.
The IBM Professor of Engineering, USC
and
(retired and now former)
Sector Vice-President & Chief Technology Officer, Northrop Grumman

(charts approved by Northrop Grumman for public release)

My *(former)* company

- ~$25 billion sales
- ~60,000 people, 50 states, 25 countries

What does an engineer do in such a setting?

- Strive to understand the customer, and how they determine value
- Create metrics for measuring your design that align with the customer’s value system
- Create engineering and management trade-space, and traverse it to create a feasible & suitable solution
- Measure the effectiveness of that solution using the metrics that resonate with the customer
- Use engineering techniques to reduce risk (“fact-based” engineering / decision-making)
- Use engineering processes to do it right the first time, and to help the team do it at scale
- Creative sense of the possible and/or feasible (technical *and* social)

A personal view

- A system designer should be a scientist, engineer, *and* artist
    - Hard science and engineering drives analysis
    - “Art” contributes to “holistic, integrated” concepts / products / systems

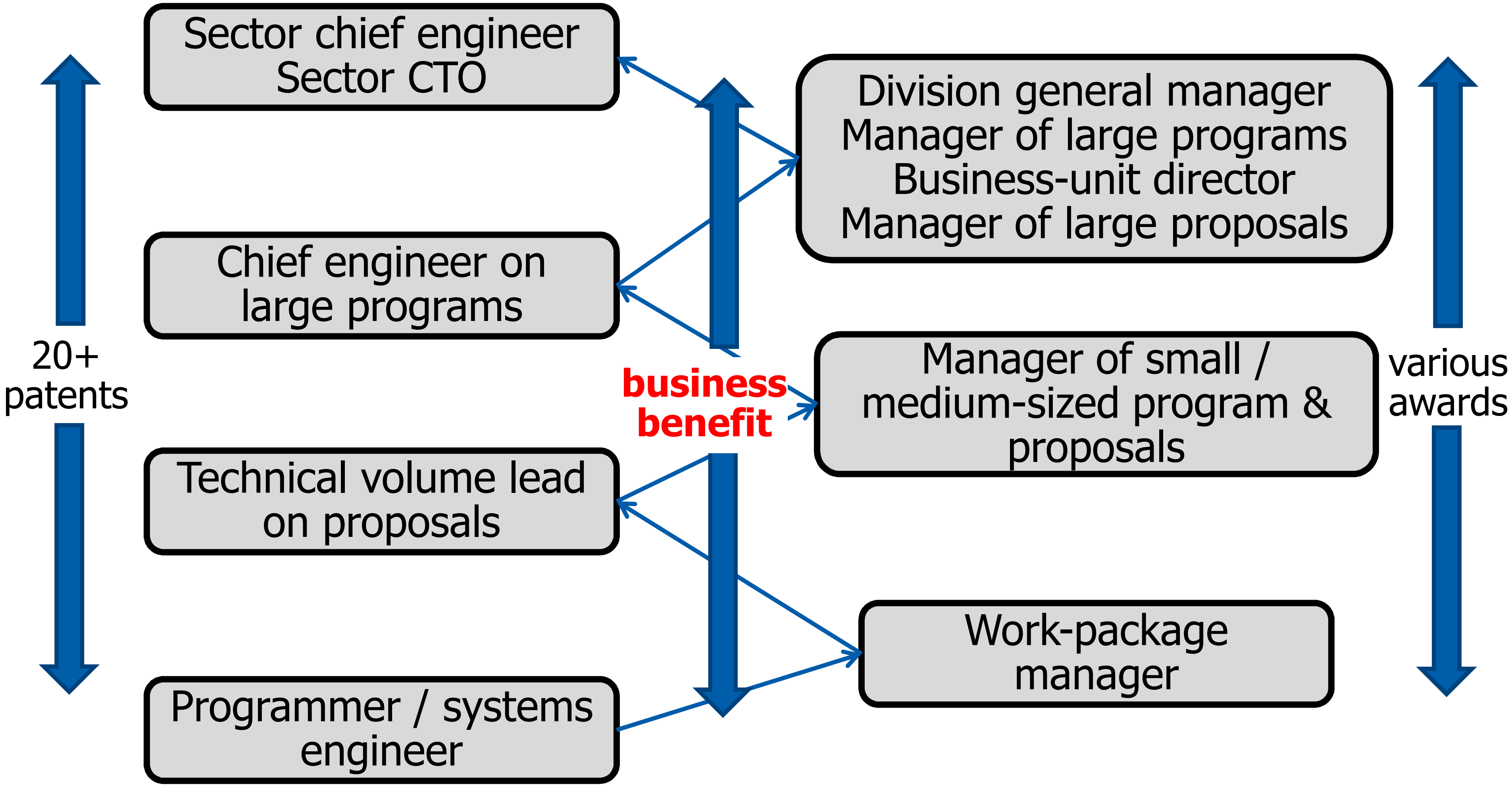
Demonstrated ability to address the social aspects of the role

- ***Leadership*** (inspiring, motivating, consensus-forming, challenging, team-building, ***accepting responsibilities***, meeting commitments, etc.)
- Ability to motivate people to want to ***work in a team*** – seeing the team's success as the path to their individual success; this can multiply your effectiveness
- Ability to translate technical skills into business success
- Constantly looking for opportunities to learn, grow, and share
- Show that you will seek out and succeed at the hard assignments
- Become an effective communicator – especially in listening and writing
- Deal with the important aspects of diversity:
  - Differing norms, differing, values, differing styles, etc.

My career

Individual technical performer

Management roles



Sector chief engineer
Sector CTO

Chief engineer on large programs

Technical volume lead on proposals

Programmer / systems engineer

20+ patents

Division general manager
Manager of large programs
Business-unit director
Manager of large proposals

Manager of small / medium-sized program & proposals

Work-package manager

various awards

business benefit

The systems architect

- Synthesizes a multi-view system concept, which is both satisfactory and feasible
- Is the conscience and driver of the effort throughout its life-cycle, with particularly strong influence in the earlier stages
  - Leads concept formulation
  - Maintains conceptual integrity throughout the process
  - Gets the team aligned around the goal and the method
  - Stays until the systems is certified for customer use, and then extracts lessons-learned from actual use (to drive the next system)
- Uses methods that are a fusion of art and science
  - Intensely creative
  - Yet grounded in rigor
- Factors will usually be in tension
  - The architect must resolve those tensions – technical and social

It’s not just about the design

- The project chief architect also:
  - Chooses the tools
  - Chooses the processes
  - Chooses the methods
  - Chooses the analysis techniques
  - Chooses the representations and views
  - Chooses the programming languages
  - Usually, chooses some of the key people
  - and so forth

. . . and is accountable for success!

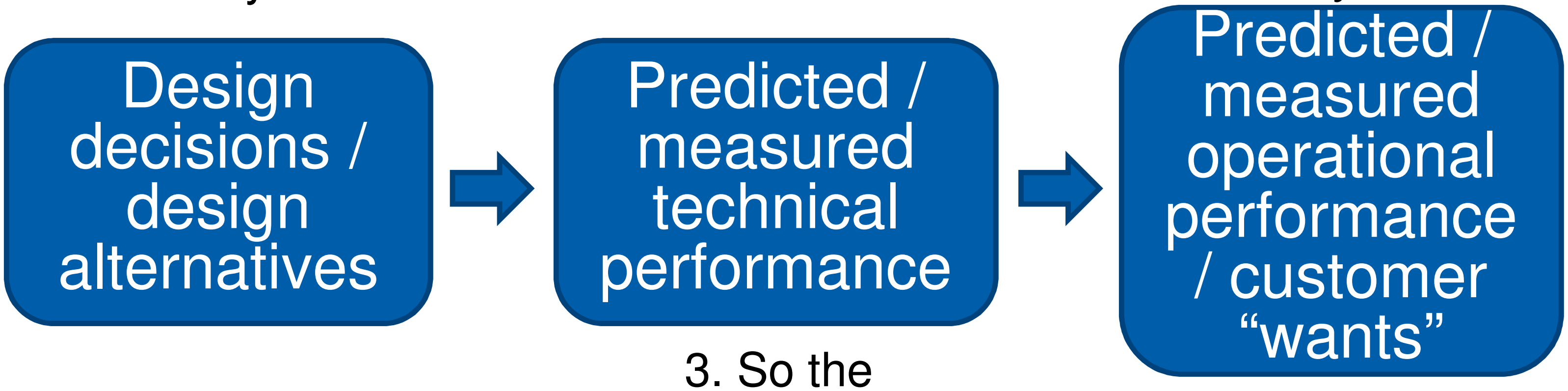
Getting started

- Define what are the characteristics of the system that your customers will value
  - Might be reliability, ease of use, easy to upgrade, fault tolerant, easy to maintain, runs fast, low cost to buy, low cost to operate and maintain . . . you have to figure that out. QFD might help.
  - Then, establish actual numeric thresholds for each
- Almost certainly, these are ***operational metrics, rather than technical ones***. They are outcomes of the design, but not the actual design alternatives that you can directly manipulate
  - So you need a mapping
  - This mapping must be transparent and credible, even to your non-technical stakeholders
- Then you can conduct trades in the design space, but assess the value of the alternatives in the operational space

Technical design must be mapped to operational benefits

1. Your degrees-of-design-freedom are mostly here

2. Your ability to demonstrate a value proposition is mostly here



Design decisions / design alternatives

Predicted / measured technical performance

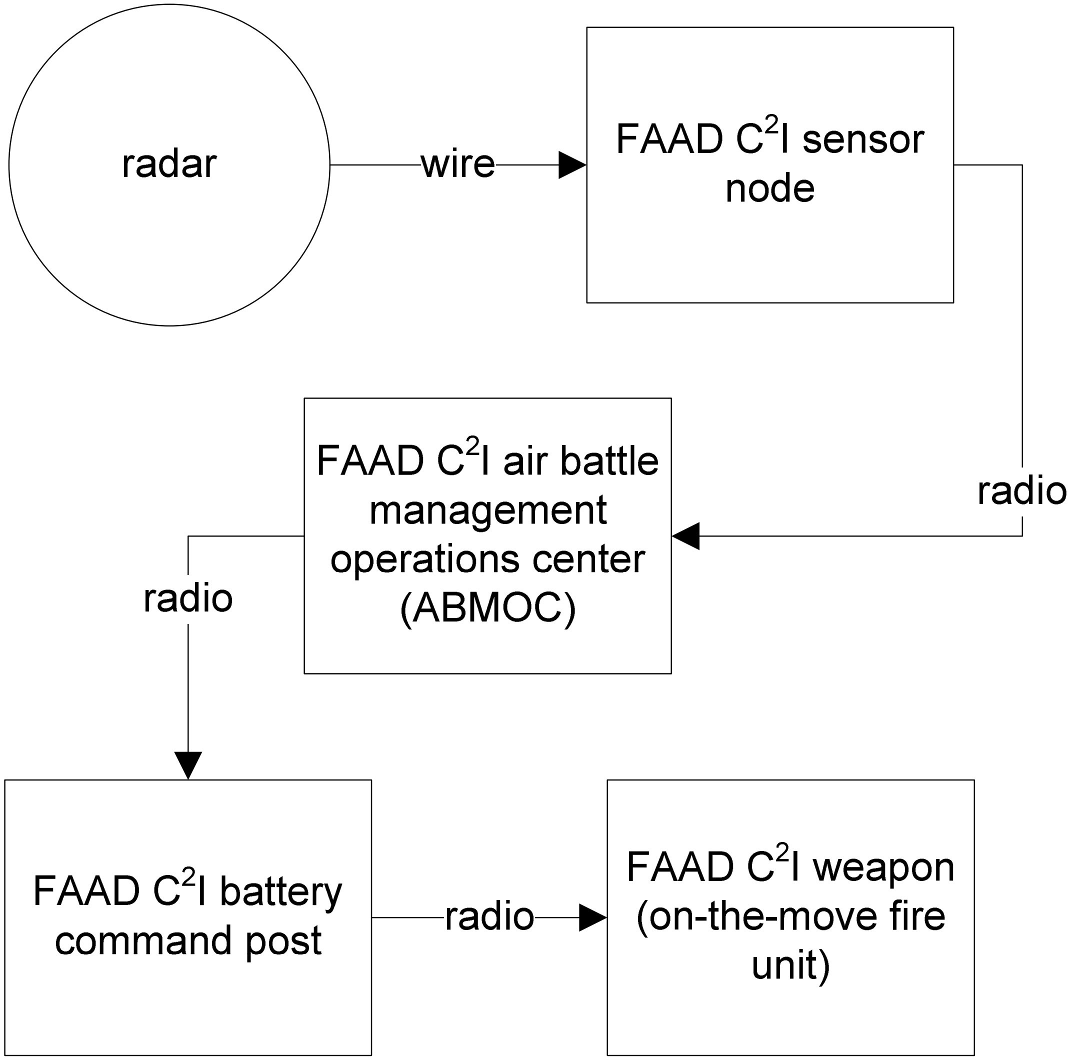
Predicted / measured operational performance / customer “wants”

3. So the mapping from the one to the other must be transparent and credible – credible even to non-technical stakeholders

Case study: How did I manage an actual design?

- Via an *end-to-end error budget* *(this is one of N things that we did)*
  - I was once the chief architect of an air defense system. We had to get the target in the narrow field-of-view of the optic 90% of the time (on the move!).
  - Given a statistical description of the slant range from the optic, that resulted in an “error basket”, for example, 5 meters in x, y , or z
  - Now, follow the entire mission thread from beginning to end, and document every source of error that will influence the accuracy:
    - The primary sensor is not where you think it is, or not pointing exactly in the direction you think it is
    - There is an error margin on the measurements made by the primary sensor
    - Between measurements, the aircraft continues to move, and our algorithm for extrapolating the position in-between actual measurements produces errors
    - Variances in port-to-port timing delays across the mission thread cause extrapolation errors
    - The fire unit is not where you think it is, or not pointing exactly in the direction you think it is
    - . . . and so forth

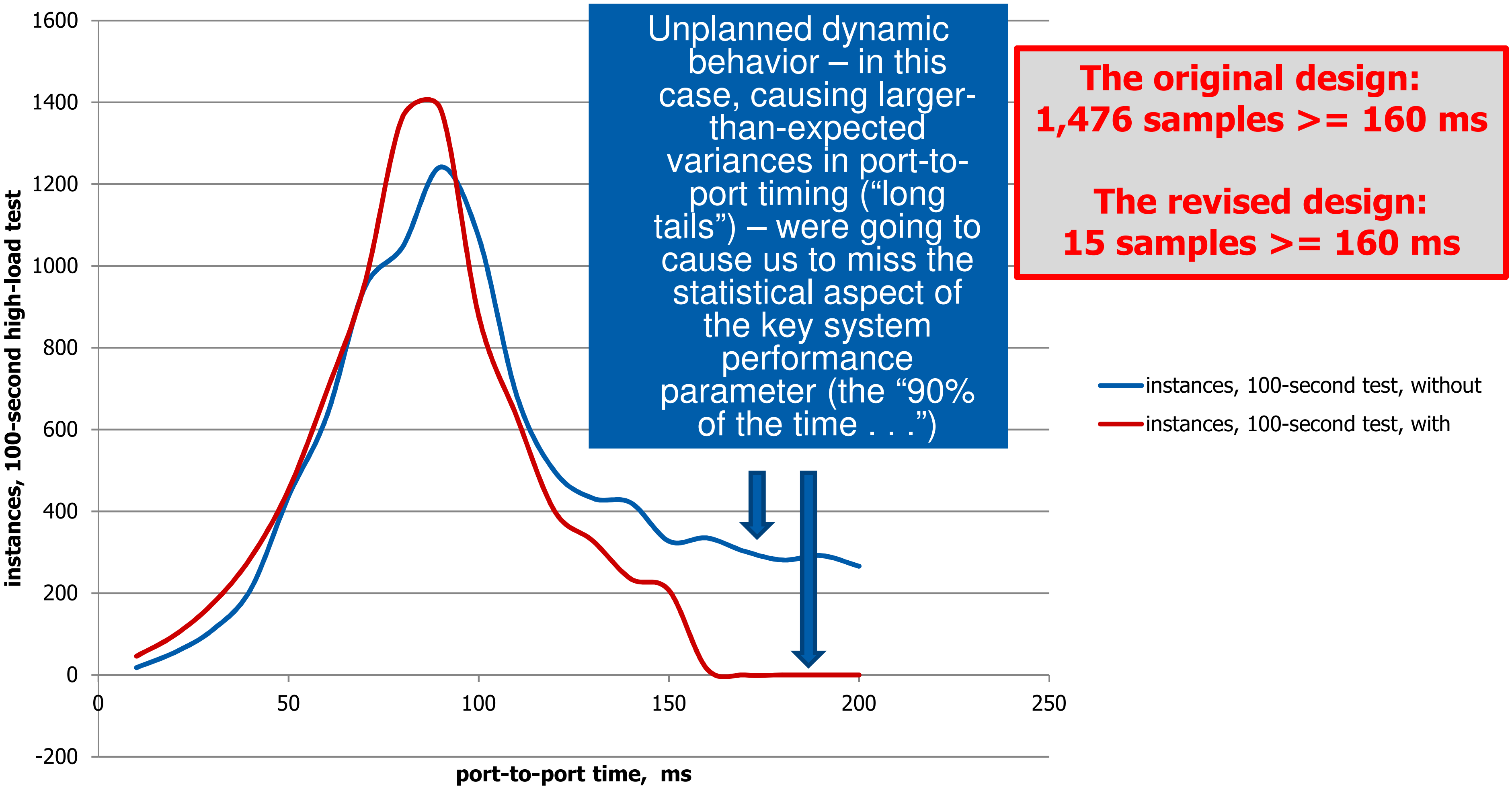
Mission threads help one understand your system


radar
wire
FAAD C^{2}I sensor node
radio
FAAD C^{2}I air battle management operations center (ABMOC)
radio
FAAD C^{2}I battery command post
radio
FAAD C^{2}I weapon (on-the-move fire unit)

Using the end-to-end error budget to manage the design

- Once we had the error model:
  - We built a model of every step along the primary mission thread
  - Supported that model with the layered models and predictive tools that related design decisions to the predicted performance and error of that step
- Calculated the error contribution of each step
- Combined the individual errors in a statistically-correct fashion, so as to create a valid statistical representation of the ultimate predicted error
- In principle, we could now understand the likely impact of every design decision
- In the real world, nothing is ever quite that simple or transparent – lots of judgment and extrapolation are required

The sort of problem that actually occurs



Unplanned dynamic behavior – in this case, causing larger-than-expected variances in port-to-port timing (“long tails”) – were going to cause us to miss the statistical aspect of the key system performance parameter (the “90% of the time . . .”)

The original design:
1,476 samples >= 160 ms

The revised design:
15 samples >= 160 ms

(about 9,000 samples in each test)

(Siegel, 2010)

What it looked like when we were done

(Northrop Grumman)

Example heuristics – Siegel

- Unplanned dynamic behavior is the true root cause of poor system reliability in most software-intensive systems, so design accordingly:
- Avoid separating control and data flows – ideally, the same entity passes both data and control. This seems to avoid an entire class of unplanned dynamic behavior.
- Software ***always*** has defects remaining. The question is how soon, and under what conditions, you start to find them.
- Far more damage is caused to program progress by bad design than by bad management . . . yet we are usually taught to ‘manage our way’ out of a problem program, rather than ‘architect our way’ out. Good management in and of itself cannot fix a bad design.
- It's never too late in the program schedule to fix a bad design.

Summary

- Being the chief architect on a complex system is a *great* job – highly creative, highly satisfying
- The attributes required to be effective in such a role are reasonably well understood, and are mostly acquired through a set of life-experiences
- Factors for which you will be called upon to make decisions will be in tension; the essence of the role is about achieving balance through a series of trade-processes
- Achieving transparency and credibility – even to your non-technical stakeholders – is vital. The need to do this drives much of what you do.
- There are lessons-learned about what typically goes wrong. Some of these are embodied in what Rechtin calls “heuristics”.