“Housekeeping”

Twitter: #ACMWebinarJudge

- Welcome to today’s ACM Webinar. The presentation starts at the top of the hour.
- If you are experiencing any problems/issues, refresh your console by pressing the F5 key on your keyboard in Windows, Command + R if on a Mac, or refresh your browser if you’re on a mobile device; or close and re-launch the presentation. You can also view the Webcast Help Guide, by clicking on the “Help” widget in the bottom dock.
- To control volume, adjust the master volume on your computer.
- If you think of a question during the presentation, please type it into the Q&A box and click on the submit button. You do not need to wait until the end of the presentation to begin submitting questions.
- At the end of the presentation, you’ll see a survey open in your browser. Please take a minute to fill it out to help us improve your next webinar experience.
- You can download a copy of these slides by clicking on the Resources widget in the bottom dock.
- This presentation is being recorded and will be available for on-demand viewing in the next 1-2 days. You will receive an automatic e-mail notification when the recording is ready.

Construx®

Software Development Best Practices

Stranger than Fiction

Case Studies in Software Engineering Judgment

www.construx.com

ACM Learning Center

http://learning.acm.org

- 1,400+ trusted technical books and videos by leading publishers including O’Reilly, Morgan Kaufmann, others
- Online courses with assessments and certification-track mentoring, member discounts on tuition at partner institutions
- Learning Webinars on big topics (Cloud/Mobile Development, Cybersecurity, Big Data, Recommender Systems, SaaS, Agile, Machine Learning, NLP, Hadoop, Parallel Programming, etc.)
- ACM Tech Packs on top current computing topics: Annotated Bibliographies compiled by subject experts
- Popular video tutorials/keynotes from ACM Digital Library, A.M. Turing Centenary talks/panels
- Podcasts with industry leaders/award winners

“Housekeeping”

- If you are experiencing any problems/issues, refresh your console by pressing the F5 key on your keyboard in Windows, Command + R if on a Mac, or refresh your browser if you’re on a mobile device; or close and re-launch the presentation. You can also view the Webcast Help Guide, by clicking on the “Help” widget in the bottom dock.

- To control volume, adjust the master volume on your computer.

- If you think of a question during the presentation, please type it into the Q&A box and click on the submit button. You do not need to wait until the end of the presentation to begin submitting questions.

- At the end of the presentation, you’ll see a survey open in your browser. Please take a minute to fill it out to help us improve your next webinar experience.

- You can download a copy of these slides by clicking on the Resources widget in the bottom dock.

- This presentation is being recorded and will be available for on-demand viewing in the next 1-2 days. You will receive an automatic e-mail notification when the recording is ready.

Talk Back

- Use the Facebook widget in the bottom panel to share this presentation with friends and colleagues

- Use Twitter widget to Tweet your favorite quotes from today's presentation with hashtag **#ACMWebinarJudge**

- Submit questions and comments via Twitter to @acmeducation – we're reading them!

Copyright Notice

Roadmap

- Judgment (Bloom’s Taxonomy)
- Using The Four Factors Model to Support Judgment
- Case Studies in Applying Software Engineering Judgment

Goal: Bring attention to a weak area in software professionalism, and introduce a means of addressing it

Judgment

```
class Class1 {

    /// <summary>
    /// The main entry point for the application.
    /// </summary>
    [STAThread]
    static void Main(string[] args) {

        // Logon
        BTVLicenseManager licenseManager = new BTVLicenseManager();

        // put your valid license here
        string networkLicense = "0[illegible]-00000000000-00000000";
        string password;

        password = "";
        licenseManager.Logon( networkLicense, password );

        Console.WriteLine( "Logged on." );

        string  fullName = @"C:\Documents and Settings\rkuo.SNAPSTREAM\My Documents\My Videos\South Park-(Freak Strike)-2004-08-17-0.mpg";

        BTVLibrary library = new BTVLibrary();

        // Get properties
        PVSPropertyBag bag = library.GetMediaByFullName( fullName );

        // Print properties to the console
        Console.WriteLine( "Properties of {0}", fullName );
        foreach( PVSProperty prop in bag.Properties ) {
            Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
        }

        // Put the PVSPropertyBag into a more friendly collection class.
        // It's a good idea for you to write a friendlier wrapper class that
        // would allow you to add and remove properties and cast back to
        // the PVSPropertyBag type on the fly.
        ArrayList aProperties = new ArrayList( bag.Properties );

        // Change the "EpisodeDescription" property
        foreach( PVSProperty prop in aProperties ) {
            if( prop.Name == "EpisodeDescription" ) {
                prop.Value = "The boys compete to appear on a talk show. (Edited by Beyond TV Framework)";
            }
        }

        // Create a new PVSPropertyBag with the edited property
        PVSPropertyBag newBag = new PVSPropertyBag();
        newBag.Properties = (PVSProperty[])aProperties.ToArray( typeof(PVSProperty) );

        // This method will edit the recording
        library.EditMedia( fullName, newBag );

        // Print properties to the console and verify the change
        Console.WriteLine( "Edited properties of {0}", fullName );
        foreach( PVSProperty prop in bag.Properties ) {
            Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
        }

        // Pause so you can see the output, hit enter to continue
        Console.WriteLine( "Press any key to exit..." );
        Console.ReadLine();
        return;
    }
}
}
```

Judgment and Bloom’s Taxonomy

Bloom’s Taxonomy

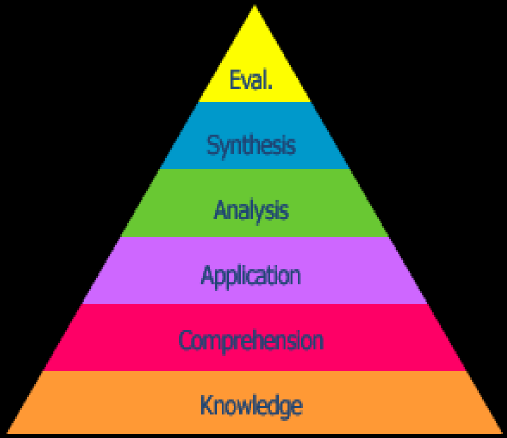
- Knowledge (Recall)
- Comprehension
- Application
- Analysis
- Synthesis (Create)
- Judgment (Evaluation)

Bloom’s Taxonomy

- Most often used in educational settings for instruction and assessment purposes
- Often described superficially or even flippantly, but a genuine understanding of Bloom’s taxonomy, especially the upper levels of the taxonomy, has **profound implications** for software professionals

Knowledge (Recall)

The remembering of previously learned material



Examples in software engineering include:

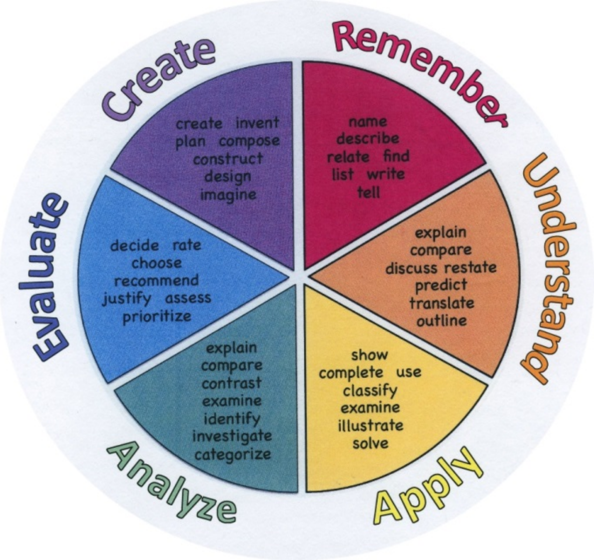
- Recall book learning
- Recall personal experience
- Remember details of technical practices
- Recall patterns of practices
- Recall successes in design, code, test, project management, and so on

Comprehension

Grasping the meaning of material

Examples in software engineering include

- Summarize a methodology, e.g., Scrum
- Explain Scrum either in words or as a diagram
- Describe an example of Scrum
- Explain why Scrum is not a design approach
- Explain how Scrum is different from Extreme Programming



Remember
name describe relate find list write tell
Understand
explain compare discuss restate predict translate outline
Apply
show complete use classify examine illustrate solve
Analyze
explain compare contrast examine identify investigate categorize
Evaluate
decide rate choose recommend justify assess prioritize
Create
create invent plan compose construct design imagine

This is the **lowest level** of **Understanding**

Application

Use of knowledge to solve problems in new and concrete situations

Examples in software engineering include

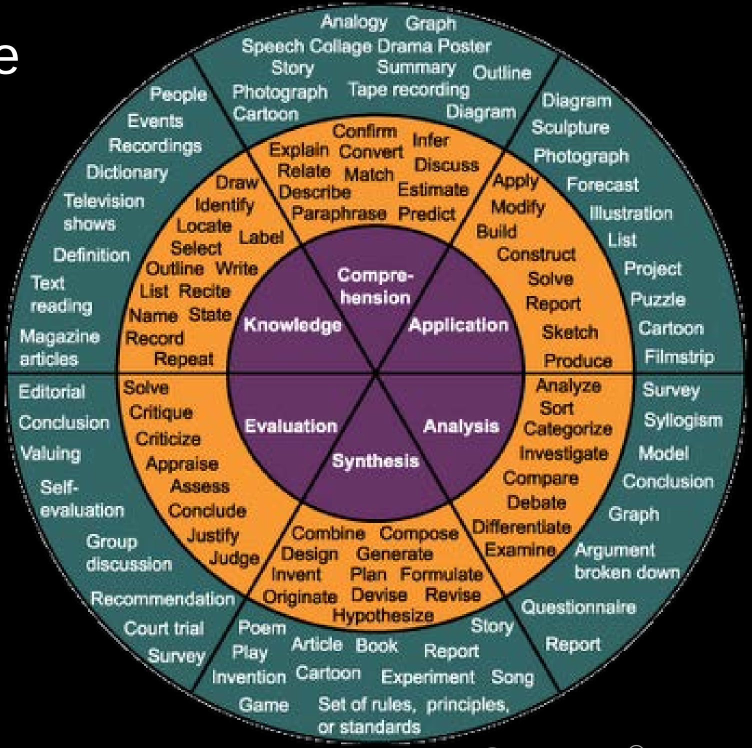
- Use general design knowledge to solve a specific design problem
- Use general project planning knowledge to plan a specific project
- Use general software construction knowledge to write a specific piece of code

Analysis

Breaking a problem into its parts so that its relationships and organizational structure can be understood

Examples in software engineering include

- ❖ Breaking a large class into two smaller classes
- ❖ Breaking a class into methods and data
- ❖ Allocating functionality and data to methods within a class
- ❖ Finding flaws in a proposed design
- ❖ Finding the source of a coding error



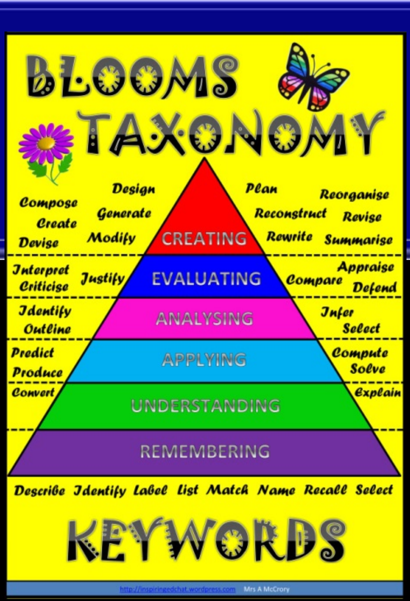
Let’s Dwell on Analysis for a Moment …

- This is also known as **Critical Thinking**
- We **screen for Analysis** skills as an entry criteria for entering the programming profession
  - Identifying the correct sequence of operations in a section of code
  - Identifying boundary conditions
  - Etc.
- These are **not common** human skills
- Result: Most software professionals are really, really good at Analysis

More on Analysis

- Analysis is an **over-developed** muscle for many technical staff
  - “Developed” is fine
  - “Over-developed” means out of balance with Synthesis and Evaluation
- Over-developed Analysis skill can lead to **Analysis Paralysis**
- Over-developed Analysis skill leads to **excessive focus** on **individual details** (inability to see the forest for the trees)

Synthesis (Create)



Putting parts together to form a new organization or whole that requires original or creative thinking

Examples in software engineering include

- Combining two classes into a new class that provides an interface at a different level of abstraction
- Making global vs. local tradeoffs in design of a system to create a better overall design
- Assembling a team based on strengths and weaknesses of a particular set of individuals
- Adjusting overall project plans based on progress of a set of individual teams

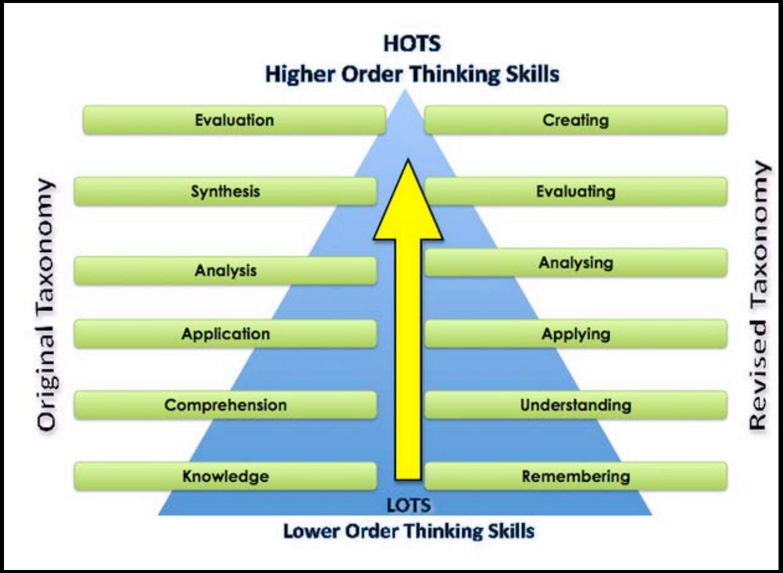
Synthesis is one of the **highest levels** of **Understanding**

More Comments about Synthesis

- This is also known as **Creative Thinking**
- This is a higher level skill, and not as many people are good at it
- Technical people often discount the value of Synthesis, e.g., technical staff's **skepticism of upper management,** which by its nature must be more focused on Synthesis/Creation than on Analysis
- The software industry does a much better job of recruiting for Analysis skill than for Synthesis/Creative skill

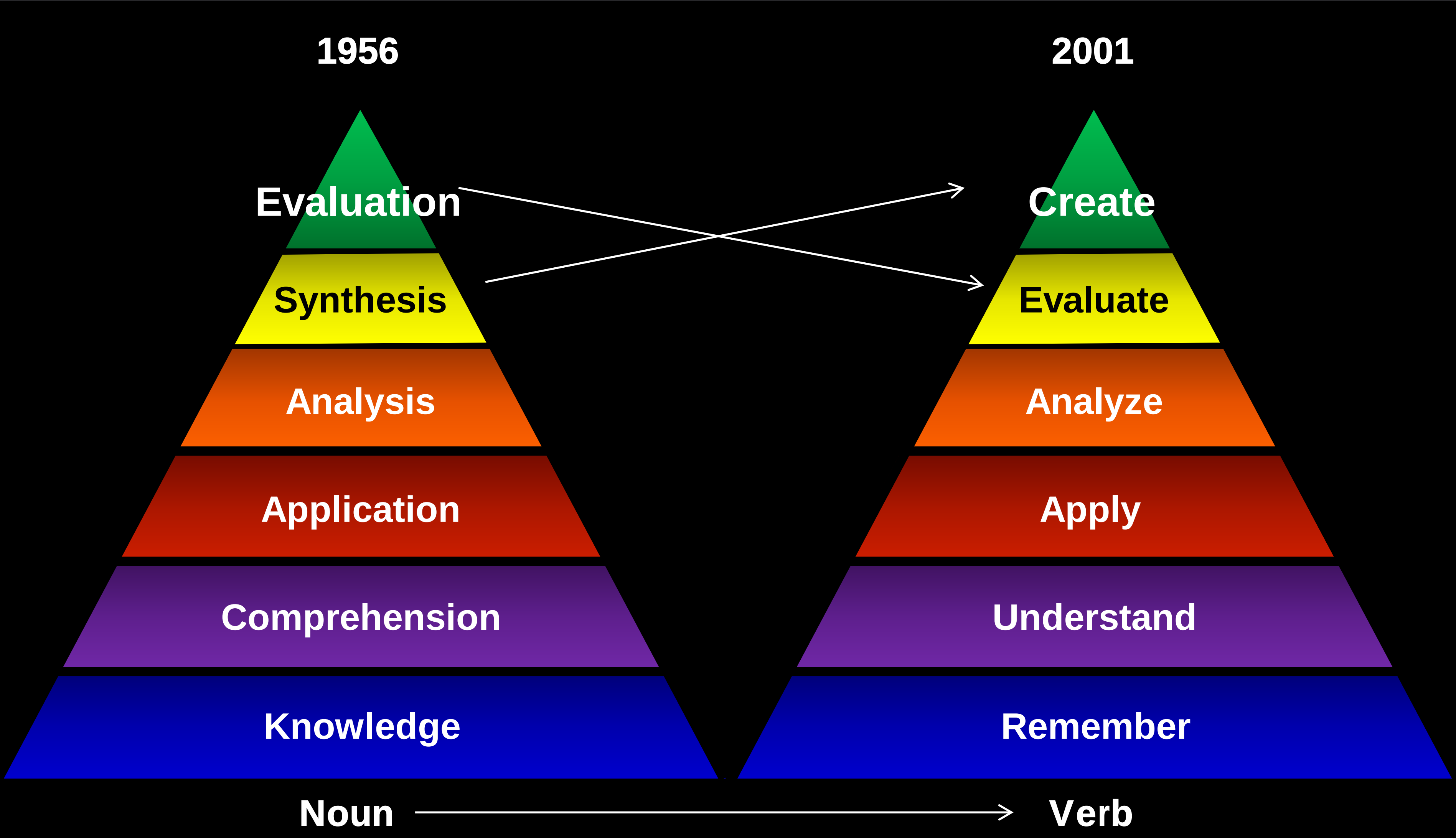
Judgment (Evaluation)

Evaluate the value of ideas, concepts, principles or solution methods for a given purpose



Like Synthesis, **Evaluation** is also one of the **highest levels** of **Understanding**

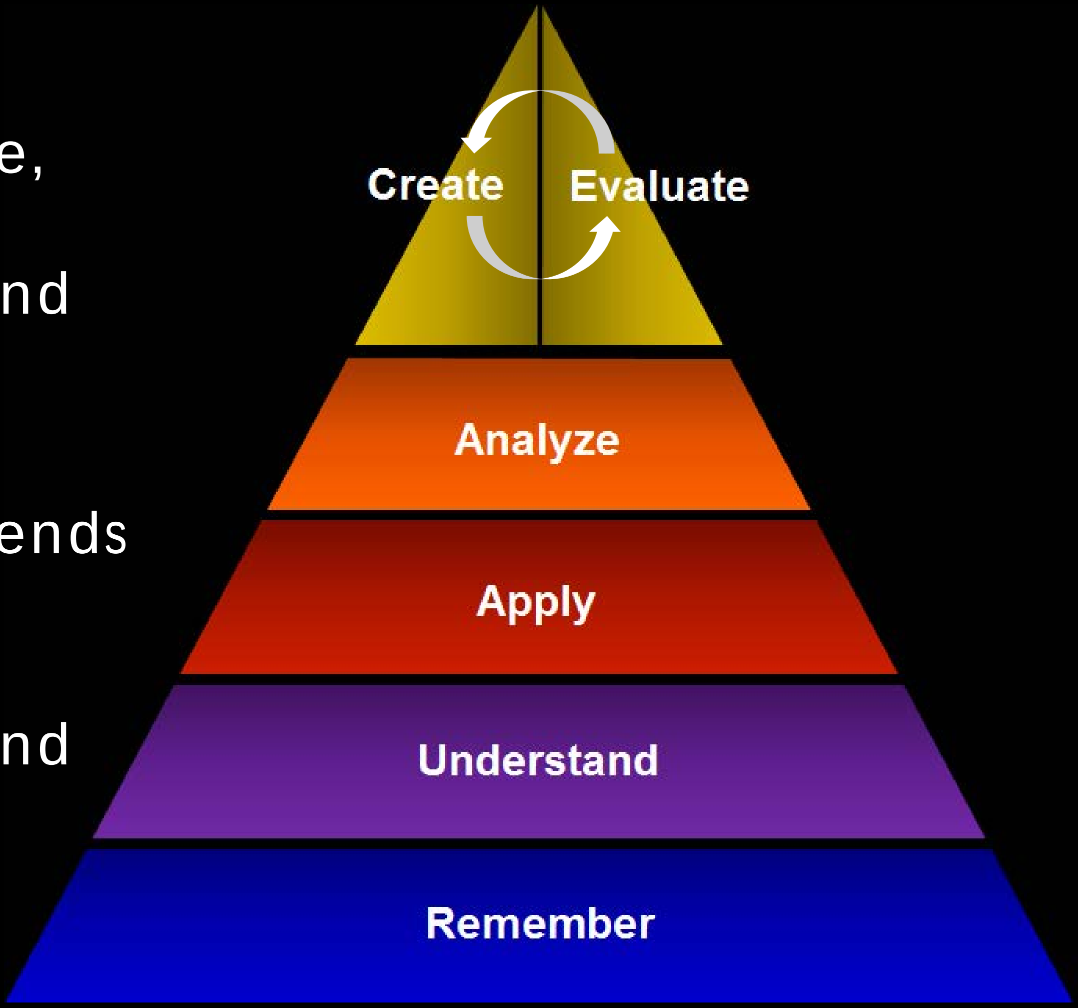
Judgment Applied to Bloom's Taxonomy



More Comments About Judgment (Evaluation)

Evaluate (Judgment) depends on Knowledge, Comprehension, Application, Analysis, and **Create (Synthesis)**

Create (Synthesis) depends on Knowledge, Comprehension, Application, Analysis, and **Evaluate (Judgment)**

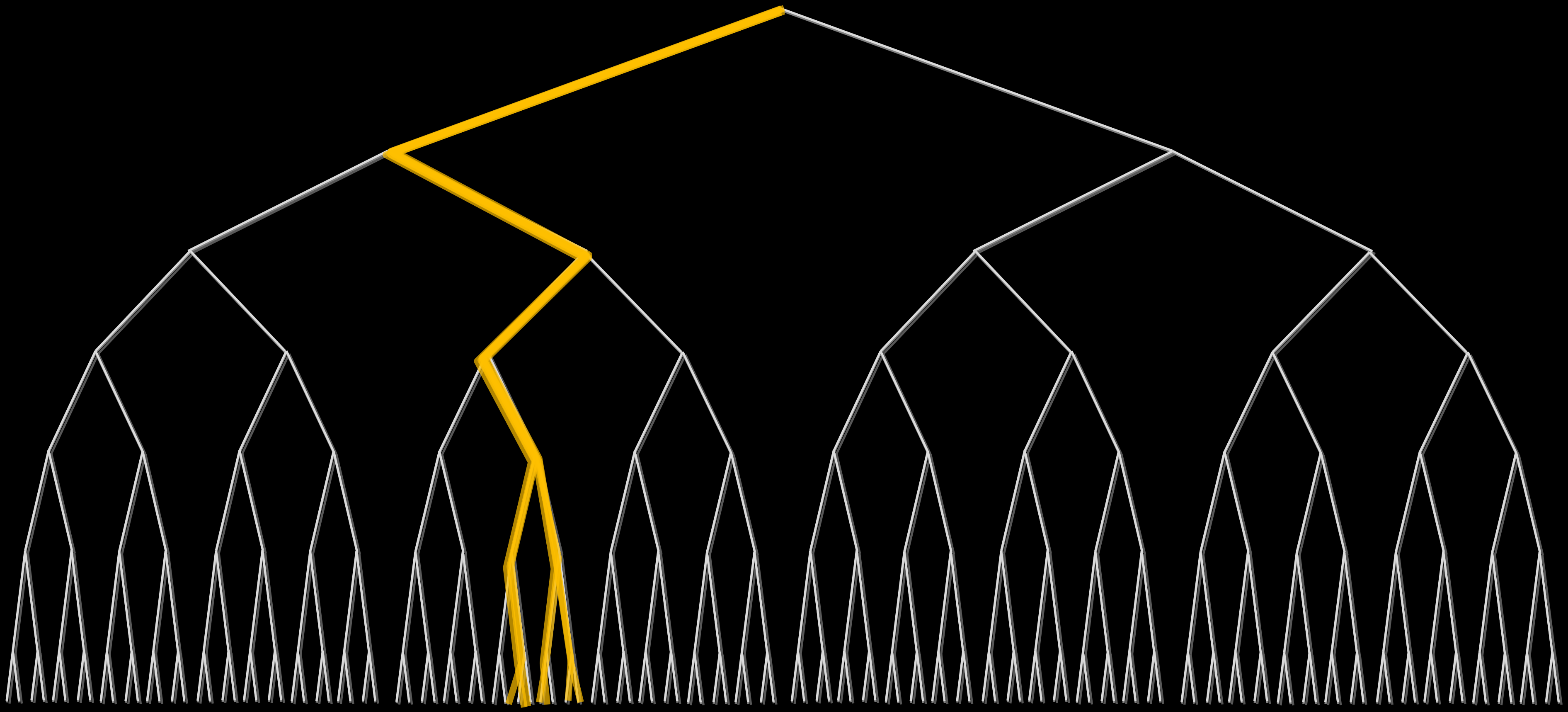


Examples of Judgment in Software Engineering

- Choose the better of two technology paths
- Choose the best of three design approaches
- Justify a re-architecture project
- Choose which proposed projects best support a business's objectives
- Assess the degree to which a new methodology will benefit an organization (or harm it)
- Predict likelihood of success of a project plan
- Conduct root cause Analysis on a failed project

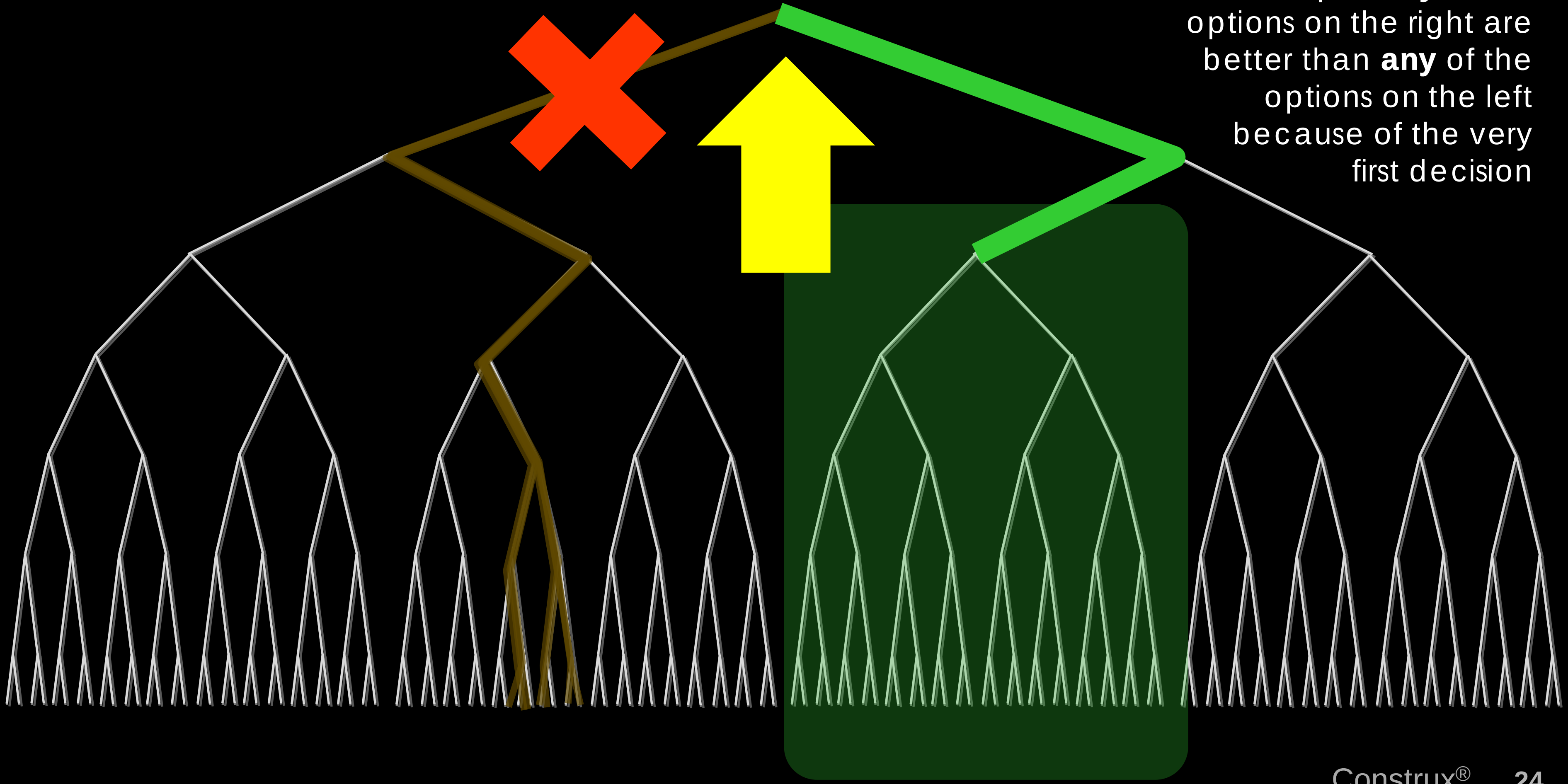
Difference Between Analysis and Judgment

Analysis is the ability to go very far down the decision tree, along multiple paths



Difference Between Analysis and Judgment

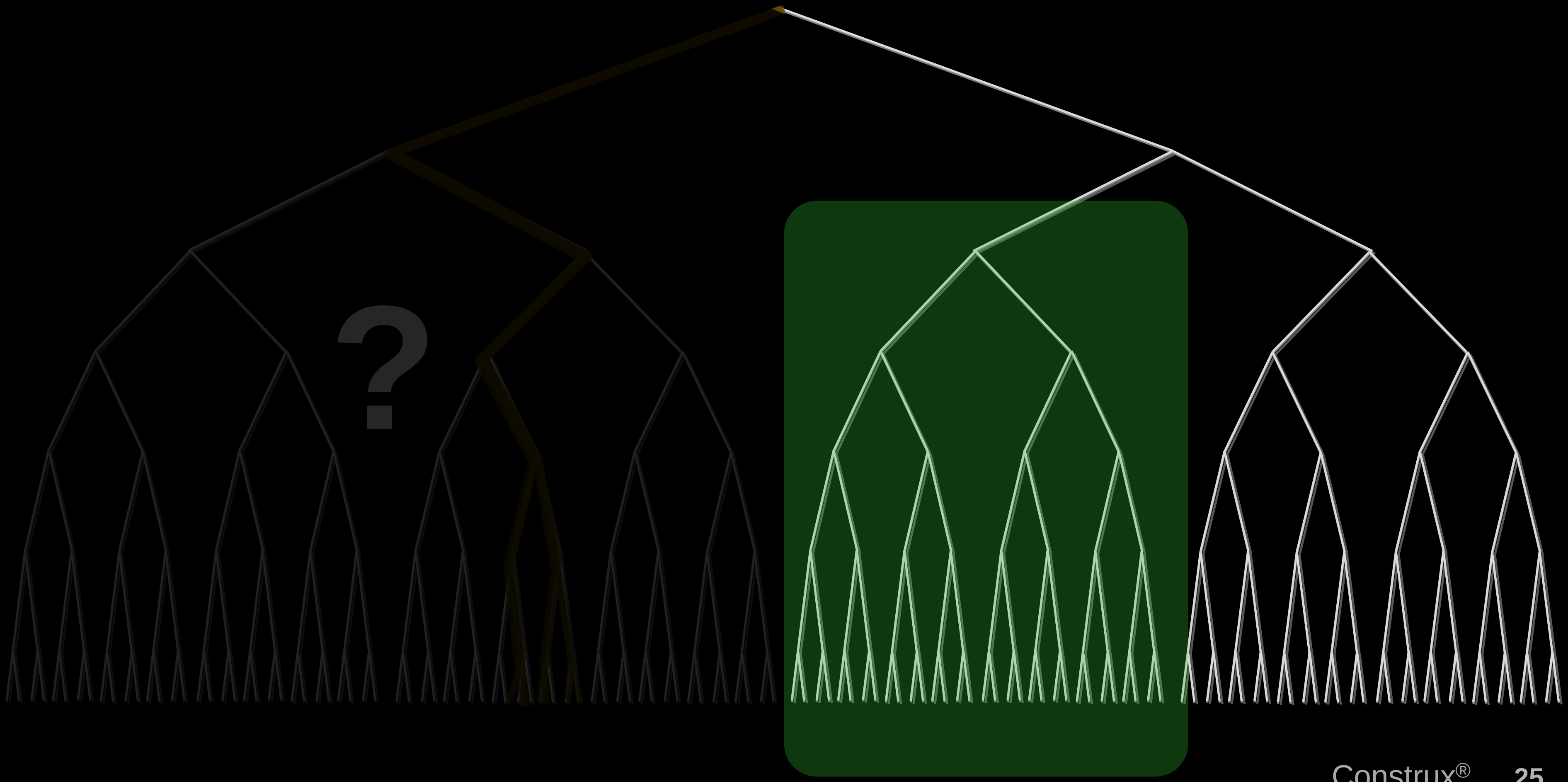
Judgment is the ability to choose the right path



For example, **any** of the options on the right are better than **any** of the options on the left because of the very first decision

Struggles with Judgment

Technical people often struggle with the idea in some cases further analysis **doesn't matter,** i.e., ignoring details



Analysis in Software is Often Mistaken for Judgment

Criticism (Analysis) in software is often mistaken for Judgment

- Criticize each of two technology paths
- Find faults in three design approaches
- Identify limitations of current system to justify a re-architecture project
- Advocacy for projects doesn't get past advocacy of one favored project
- Assessment of a new methodology amounts to a religious advocacy for one methodology
- Assessment of project plans focuses on minutia
- Root cause analysis on a failed project consists of rehashing unpopular decisions

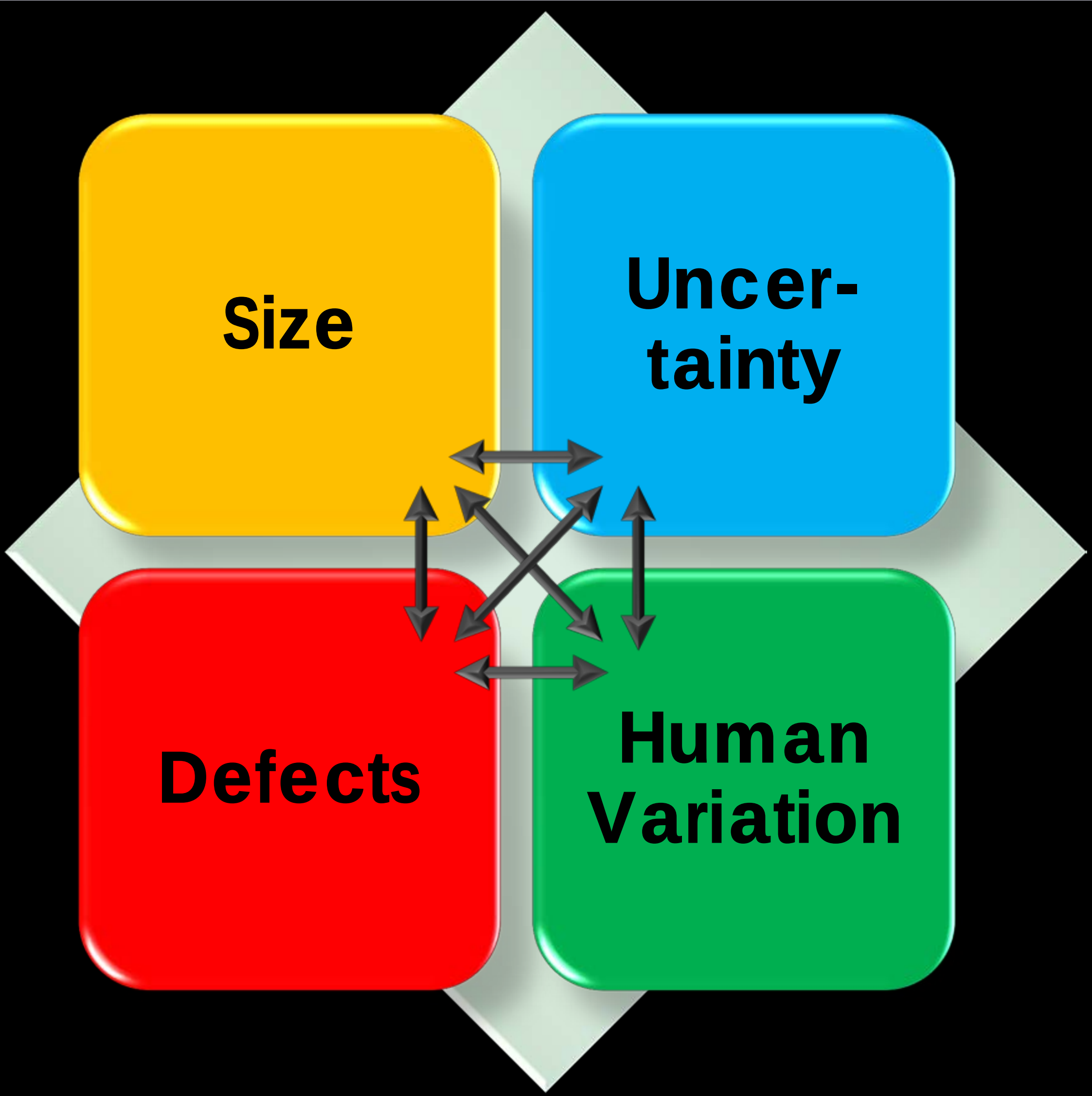
Judgment in Software Engineering

- **Judgment** capability is even rarer than **Synthesis** capability
- We hardly screen for **Judgment** in software staff **at all**
  - E.g., Microsoft's famous interview questions are nearly all about **Synthesis** (and that is higher on Bloom's Taxonomy than typical interview questions)
- **Poor Business Judgment** is so common among technical staff that it is a **cliché**
- The $64 question is, How do we **Develop Good Judgment** in Software Professionals?

The Four Factors Model

Four Factors Model

Introduced at Construx Software Executive Summit 2013



Four Factors

SIZE (diseconomy of scale; failure rate; specializations; mix of activities)

UNCERTAINTY (intellectual phases; cone of uncertainty; feature staircase vs. feature buildup; risk management; effort vs. certainty curve)

DEFECTS (DCI, defect detection lag, defect removal techniques in series, relationship to process stability)

HUMAN VARIATION (effect on research; effect on selection of methods (familiar vs. unfamiliar); effect on team composition, team cohesion, recruiting, and retention; focus on perfect execution vs. perfect plans; implication for favoring robust methods)

The Four Factors and Judgment

- The Four Factors model provides a set of **Templates** against which we compare what we see on a project vs. what we would expect to see, and that supports **Judgment**
- For example, we could create checklists based on the four factors …

Size Checklist 1/2

- □ Is the project estimated close to its actual size?
- □ Does the project's schedule permit completion of a project of the estimated effort?
- □ Is the project planned at a level commensurate with its size?
- □ Does the project have appropriate allocation of activities for its size?
- □ Does the project have appropriate staff specializations for its size?
- □ Does the project have appropriate levels of management for its size?

Size Checklist 2/2

- □ Does the project have QA practices appropriate for its size?
- □ Is the project appropriate addressing the factors that scale disproportionately with size (Precedentedness, Process Maturity, Risk Resolution, Requirements Flexibility, Team Cohesion, per Cocomo)?

Uncertainty Checklist 1/2

- □ Do the project’s estimates and plans account for the Cone of Uncertainty?
- □ Where will the project’s challenges come from in terms of the Intellectual Phase Profiles?
- □ Is requirements uncertainty addressed and manageable?
- □ Is design uncertainty addressed and manageable?
- □ Is technology uncertainty addressed and manageable?
- □ Is the degree of precedentedness manageable for the size of the project?
- □ Is planning uncertainty addressed and manageable?

Uncertainty Checklist 2/2

- □ Is the project striking an appropriate balance between time allocated for proactive activities vs. time allocated for reactive activities?
- □ Is risk management in place and appropriate for the size of the project?
- □ Is the overall level of uncertainty manageable for the size of the project?

Defect Checklist 1/1

- □ Is the project using practices that will minimize the gap between defect insertion and defect detection?
- □ Is the series of defect removal practices capable of producing the desired level of quality?
- □ Is the series of defect removal practices efficient in achieving the desired level of quality?
- □ Are the quantity and kinds of defect removal appropriate for the size of the project?
- □ Are the quantity and kinds of defect removal appropriate for the quantity and kind of uncertainty on the project?
- □ Are the quantity and kinds of defect removal appropriate for the capabilities of the people working on the project?

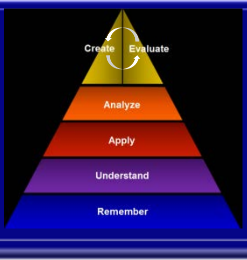
Human Variation Checklist 1/2

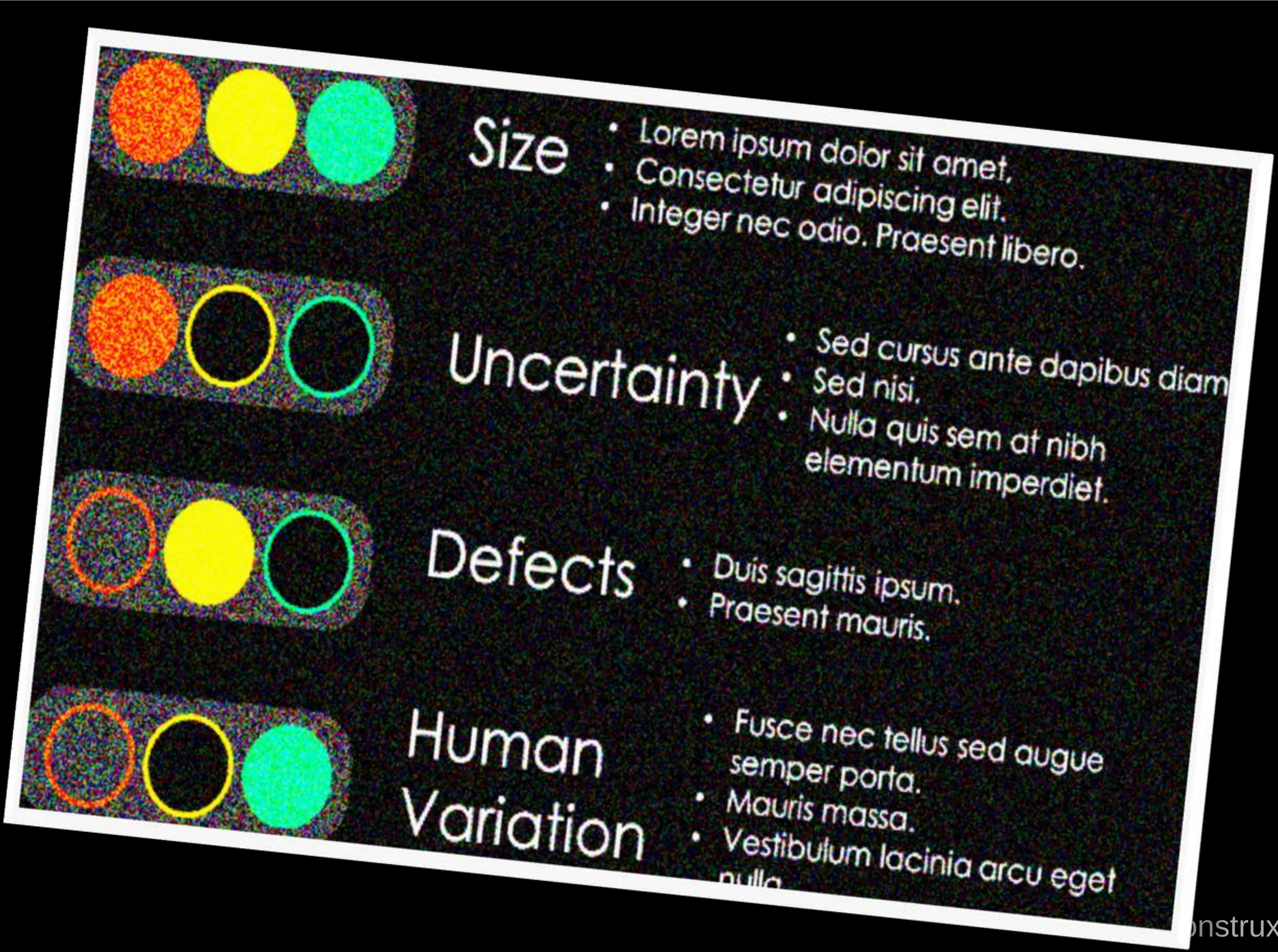
- □ Do the people on the project have the skills to complete a project of the intended size?
- □ Do the people on the project have the skills to complete a project with this project’s uncertainty characteristics?
- □ Do the people on the project have the skills to complete a project with this project’s intended quality level?
- □ Is the requirements skill level matched to both with the size of the project and degree of challenge in the requirements area?
- □ Is the design/architecture skill level matched to both with the size of the project and degree of challenge in the design area?
- □ Is the project management skill and experience matched to the project size and overall challenge?

Human Variation Checklist 2/2

- □ What is the motivation level of the people on the project?
- □ Does the level of staff turnover support a project of the intended size?
- □ Do staff capabilities support the human/staff organization of the project, including geographic distribution?
- □ Is the staff's experience in the business area suitable for the size, uncertainty level, and desired quality level of the project?
- □ Is the staff's experience in the technology platform suitable for the size, uncertainty level, and desired quality level of the project?

Simplified Application of the Checklists in this Talk





● ● ● **Size**
- Lorem ipsum dolor sit amet.
- Consectetur adipiscing elit.
- Integer nec odio. Praesent libero.

● ○ ○ **Uncertainty**
- Sed cursus ante dapibus diam
- Sed nisi.
- Nulla quis sem at nibh elementum imperdiet.

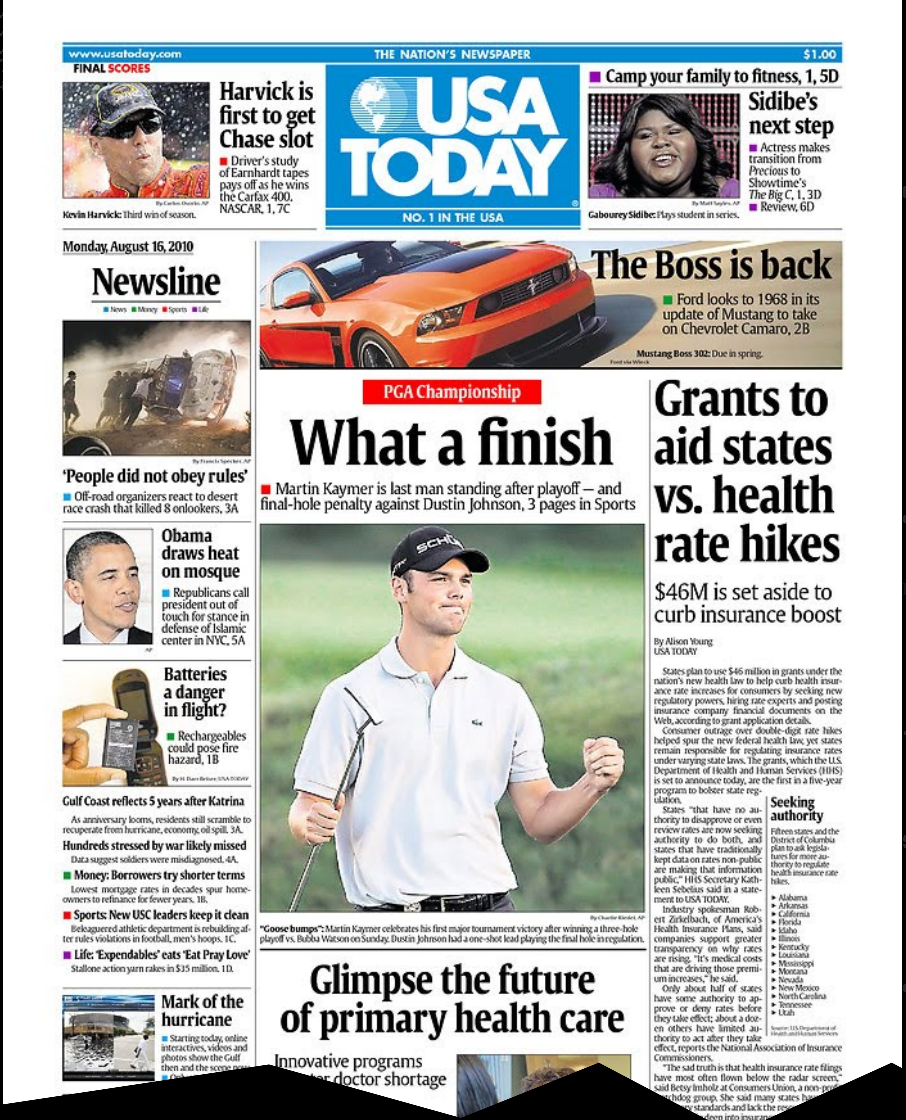
○ ● ○ **Defects**
- Duis sagittis ipsum.
- Praesent mauris.

○ ○ ● **Human Variation**
- Fusce nec tellus sed augue semper porta.
- Mauris massa.
- Vestibulum lacinia arcu eget nulla

Case Studies

Value of Case Studies

- A deep understanding of the Four Factors supports **dramatically better Software Engineering Judgment** than we usually see
- Understanding of the Four Factors supports **Synthesis/Creative** (in the Bloom's taxonomy sense) in **planning** and **management** too
- Case studies provide experience recognizing patterns, and developing and applying Judgment



www.usatoday.com THE NATION'S NEWSPAPER $1.00

FINAL SCORES

Harvick is first to get Chase slot

■ Driver's study of Earnhardt tapes pays off as he wins the Carfax 400. NASCAR, 1, 7C

By Carlos Osorio, AP

Kevin Harvick: Third win of season.

USA TODAY

NO. 1 IN THE USA

■ Camp your family to fitness, 1, 5D

Sidibe's next step

■ Actress makes transition from *Precious* to Showtime's *The Big C*, 1, 3D
■ Review, 6D

By Matt Sayles, AP

Gabourey Sidibe: Plays student in series.

Monday, August 16, 2010

Newsline

■ News ■ Money ■ Sports ■ Life

By Francis Specker, AP

'People did not obey rules'

■ Off-road organizers react to desert race crash that killed 8 onlookers, 3A

Obama draws heat on mosque

■ Republicans call president out of touch for stance in defense of Islamic center in NYC, 5A

AP

Batteries a danger in flight?

■ Rechargeables could pose fire hazard, 1B

By H. Darr Beiser, USA TODAY

Gulf Coast reflects 5 years after Katrina

As anniversary looms, residents still scramble to recuperate from hurricane, economy, oil spill. 3A.

Hundreds stressed by war likely missed

Data suggest soldiers were misdiagnosed. 4A.

■ Money: Borrowers try shorter terms

Lowest mortgage rates in decades spur homeowners to refinance for fewer years. 1B.

■ Sports: New USC leaders keep it clean

Beleaguered athletic department is rebuilding after rules violations in football, men's hoops. 1C.

■ Life: 'Expendables' eats 'Eat Pray Love'

Stallone action yarn rakes in $35 million. 1D.

Mark of the hurricane

■ Starting today, online interactives, videos and photos show the Gulf then and the scene now

The Boss is back

■ Ford looks to 1968 in its update of Mustang to take on Chevrolet Camaro, 2B

Mustang Boss 302: Due in spring.

Ford via Wieck

PGA Championship

What a finish

■ Martin Kaymer is last man standing after playoff — and final-hole penalty against Dustin Johnson, 3 pages in Sports

By Charlie Riedel, AP

"Goose bumps": Martin Kaymer celebrates his first major tournament victory after winning a three-hole playoff vs. Bubba Watson on Sunday. Dustin Johnson had a one-shot lead playing the final hole in regulation.

Glimpse the future of primary health care

Innovative programs
... doctor shortage

Grants to aid states vs. health rate hikes

$46M is set aside to curb insurance boost

By Alison Young
USA TODAY

States plan to use $46 million in grants under the nation's new health law to help curb health insurance rate increases for consumers by seeking new regulatory powers, hiring rate experts and posting insurance company financial documents on the Web, according to grant application details.

Consumer outrage over double-digit rate hikes helped spur the new federal health law, yet states remain responsible for regulating insurance rates under varying state laws. The grants, which the U.S. Department of Health and Human Services (HHS) is set to announce today, are the first in a five-year program to bolster state regulation.

States "that have no authority to disapprove or even review rates are now seeking authority to do both, and states that have traditionally kept data on rates non-public are making that information public," HHS Secretary Kathleen Sebelius said in a statement to USA TODAY.

Industry spokesman Robert Zirkelbach, of America's Health Insurance Plans, said companies support greater transparency on why rates are rising. "It's medical costs that are driving those premium increases," he said.

Only about half of states have some authority to approve or deny rates before they take effect; about a dozen others have limited authority to act after they take effect, reports the National Association of Insurance Commissioners.

"The sad truth is that health insurance rate filings have most often flown below the radar screen," said Betsy Imholz at Consumers Union, a non-profi... watchdog group. She said many states ha... ...ry standards and lack the res... ...deep into insuran...

Seeking authority

Fifteen states and the District of Columbia plan to ask legislatures for more authority to regulate health insurance rate hikes.

- Alabama
- Arkansas
- California
- Florida
- Idaho
- Illinois
- Kentucky
- Louisiana
- Mississippi
- Montana
- Nevada
- New Mexico
- North Carolina
- Tennessee
- Utah

Source: U.S. Department of Health and Human Services

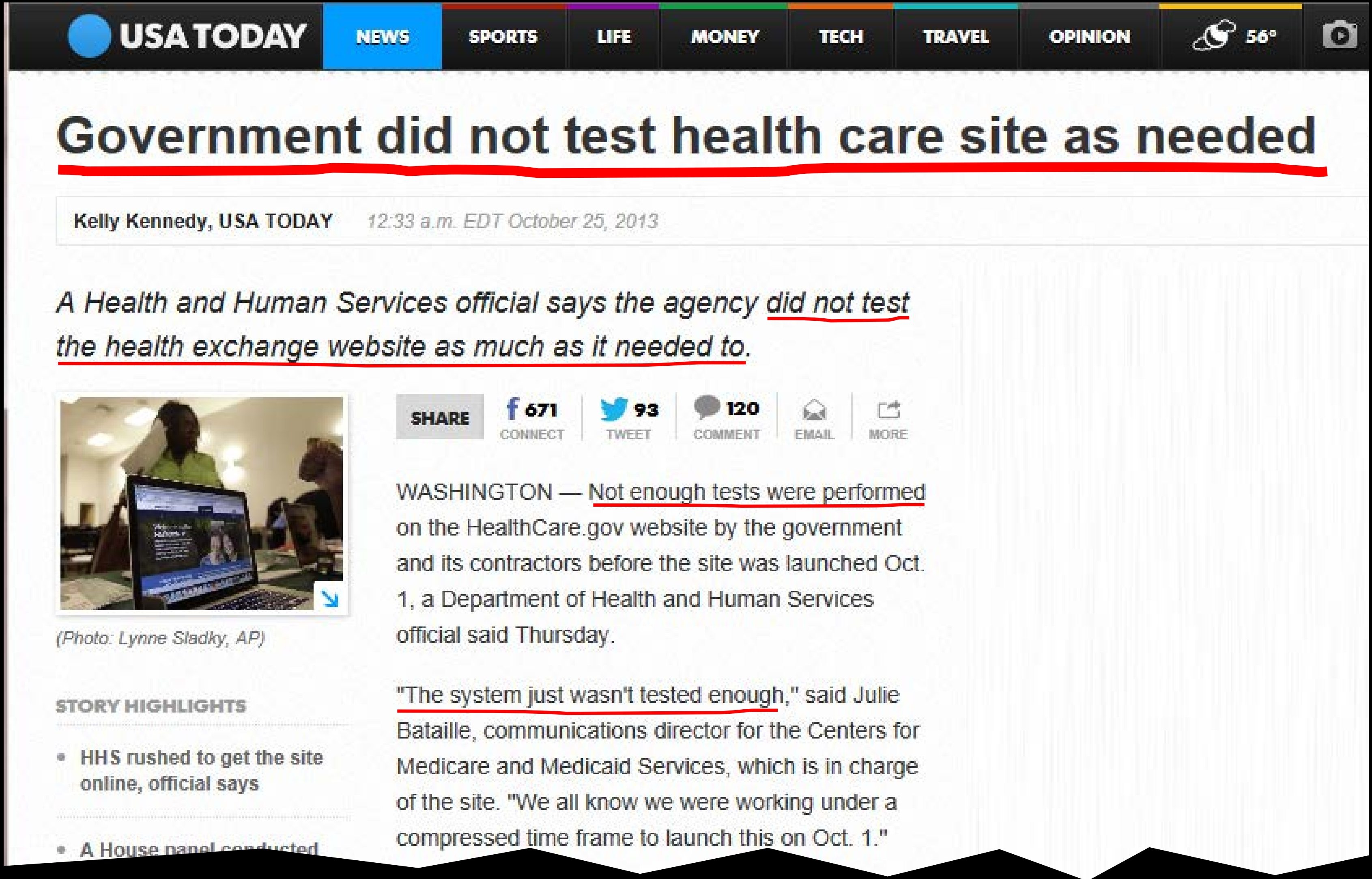
Healthcare.gov

2013

Healthcare.gov Background

- Affordable Care Act passed December 2009, signed into **law** in **March 2010**
- Private sector development **contracts awarded in 2011**
- Original project budget was about $100 million
- **Coding** by CGI (prime contractor) **began** in **Spring 2013** for **October 1, 2013** “go live” date
- Cost by the time the system went live was almost $300 million
- When the system went live it was plagued by slow performance, down time, lost data, incomplete functionality, and other problems—one estimate was that only 1% of people were able to use the site as intended at first

How Did the People Involved with Healthcare.gov Diagnose the Problems?



Government did not test health care site as needed

Kelly Kennedy, USA TODAY 12:33 a.m. EDT October 25, 2013

A Health and Human Services official says the agency did not test the health exchange website as much as it needed to.

(Photo: Lynne Sladky, AP)

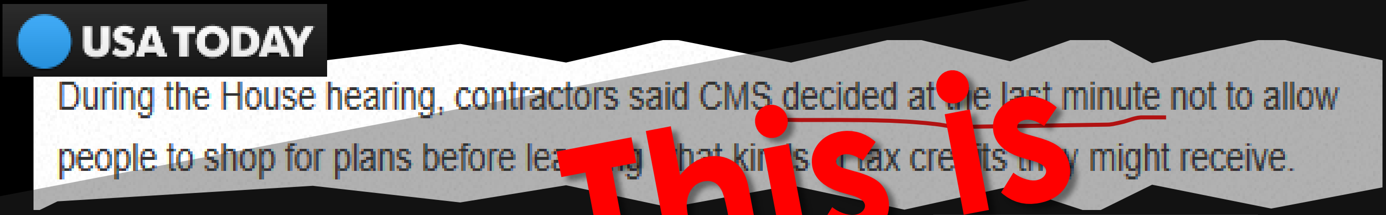
STORY HIGHLIGHTS

- HHS rushed to get the site online, official says
- A House panel conducted

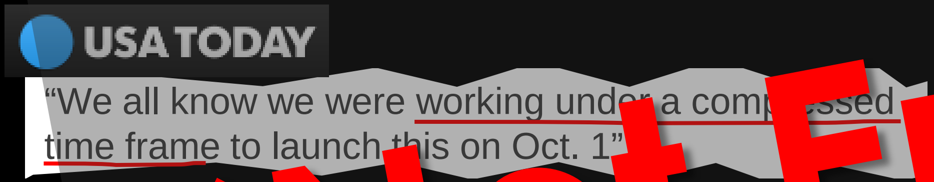
WASHINGTON — Not enough tests were performed on the HealthCare.gov website by the government and its contractors before the site was launched Oct. 1, a Department of Health and Human Services official said Thursday.

"The system just wasn't tested enough," said Julie Bataille, communications director for the Centers for Medicare and Medicaid Services, which is in charge of the site. "We all know we were working under a compressed time frame to launch this on Oct. 1."

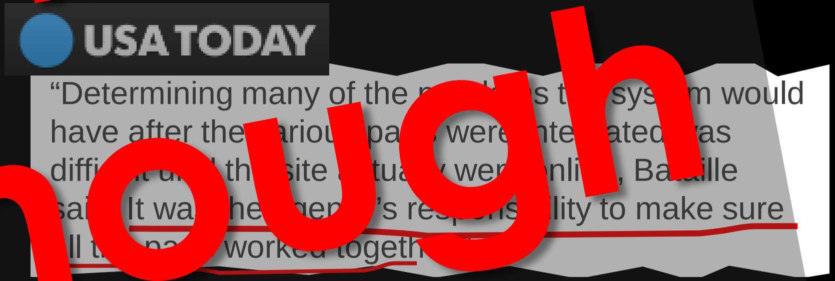
Other Details About Healthcare.gov


USA TODAY

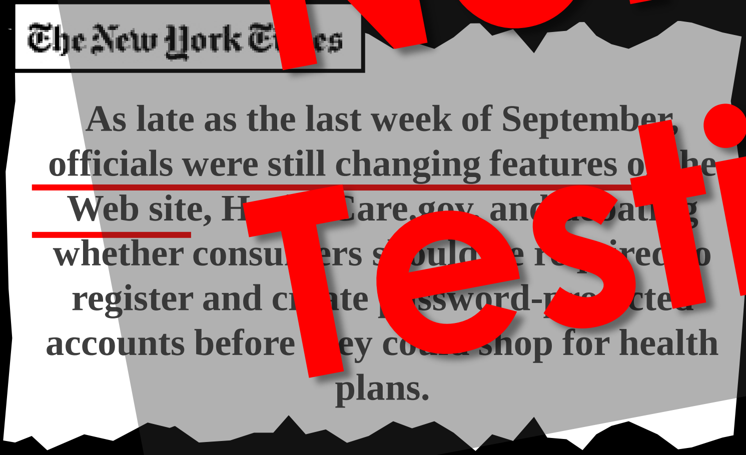
During the House hearing, contractors said CMS decided at the last minute not to allow people to shop for plans before learning what kinds of tax credits they might receive.


USA TODAY

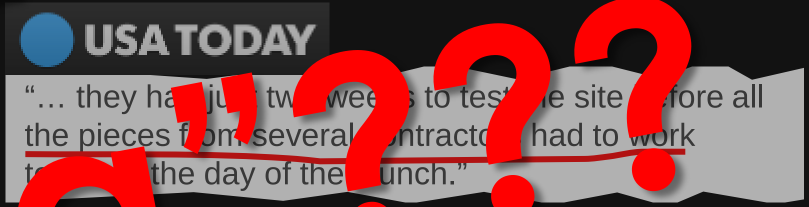
"We all know we were working under a compressed time frame to launch this on Oct. 1"


USA TODAY

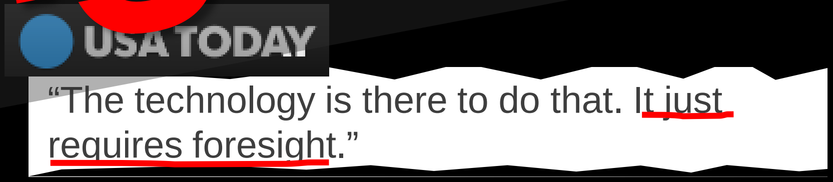
"Determining many of the problems the system would have after the various parts were integrated was difficult until the site actually went online," Bataille said. It was the agency's responsibility to make sure all the parts worked together.


The New York Times

As late as the last week of September, officials were still changing features of the Web site, HealthCare.gov, and debating whether consumers should be required to register and create password-protected accounts before they could shop for health plans.


USA TODAY

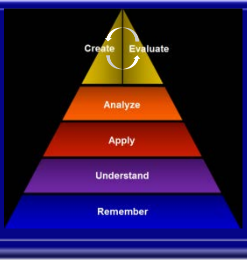
"… they had just two weeks to test the site before all the pieces from several contractors had to work together the day of the launch."


USA TODAY

"The technology is there to do that. It just requires foresight."

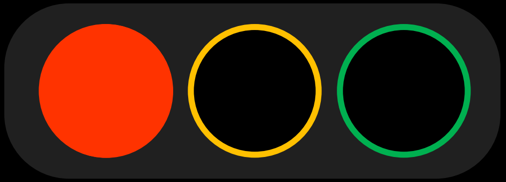
This is "Not Enough Testing"???

The Four Factors Model Applied to Healthcare.gov in 2013



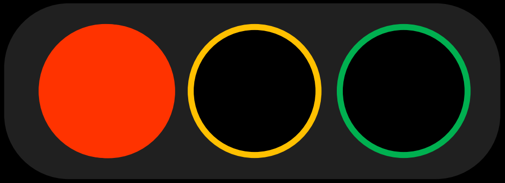
Size

- Short schedule
- Huge budget
- Huge staff ramp-up
- Planning not matched to project size



Uncertainty

- Numerous immoveable requirements (laws)
- Massive requirements changes
- Significant unprecedentedness



Defects

- Approach to QA not matched to size of project or nature of uncertainty

Human Variation

- Does not matter!

Update: GAO Report **July 2014**

Healthcare.gov suffered from

- Rushed schedule
- Changing requirements
- Lax oversight of contractors
- Lack of effective planning and oversight practices

Evaluation in the July 2014 GAO Report is substantially similar to the evaluation I gave in November 2013 (at Construx's 2013 Software Executive Summit) just from reading the newspaper

*I believe **anyone can do this** if they understand the Four Factors Model*

Construx®

Software Development Best Practices

```
class Class1 {
```

SPORTS

Blazers feeling loose and motivated, look to extend win streak **B1**

Former Ducks coach Ernie Kent is hired by Washington State **B1**

LIVING

New 'anonymity apps' let you post secrets — but the idea isn't so new, and they're not so anonymous **A8**

The Oregonian

$1.00 ALWAYS ON OREGONLIVE.COM TUESDAY, APRIL 1, 2014

Old map led to overlogging

By Ken Armstrong, Justin Mayo and Mike Baker
Seattle Times

SEATTLE – State regulators have been using outdated boundaries to restrict logging above the Snohomish County slope that collapsed March 22, failing to incorporate newer research that would have protected a swath of land that wound up being clear-cut, according to a Seattle Times analysis of documents and geographical data.

Because trees intercept and absorb water, removing them can contribute to the risk or size of a landslide by lowering the soil's saturation point, according to geological reports. The impact can linger for years.

In 1997, a report commissioned by the state Department of Ecology used "newly developed computational tools" to map the plateau atop the unstable hill outside Oso. That report was prepared by geologist Daniel J. Miller and hydrologist Joan Sias; Miller's portion drew boundaries for where groundwater could feed into the slope and increase the risks of landslide.

When the Department of Natural Resources issued logging restrictions later that same year, the agency cited the Miller-Sias report and treated it as state of the art, saying any future study should emulate its methods. But instead of adopting Miller's map, the DNR used boundaries that had been drawn up in 1988.

"We did the work. It was cited in the prescriptions as what you should do. And it appears

*See **Slide**, A4*

Latest developments on Oso landslide

Official death toll: 24, and 17 of those victims have been positively identified.

Still missing: 22.

Search details: More is known about the force of the slide, which helps better find victims in a debris field that is 70 feet deep in places. Steve Harris, a division supervisor, said dogs are still the primary search tool. Human remains are found four to six times a day, but sometimes only partial remains, which makes identification harder.

Progress: A makeshift road completed over the weekend links one side of the 300-acre debris field to the other.

Financial losses: Now estimated at $10 million, Gov. Jay Inslee said Monday as he sought a federal major-disaster declaration. About 30 families need help with housing and personal and household goods.

– Associated Press

One last, mad dash for health coverage

By Ricardo Alonso-Zaldivar
Associated Press

WASHINGTON – In a flood of last-minute sign-ups, hundreds of thousands of Americans rushed to apply for health insurance Monday, but deadline day for President Barack Obama's overhaul brought long, frustrating waits and a new spate of website ills.

"This is like trying to find a parking spot at Walmart on Dec. 23," said Jason Stevenson, working with a Utah nonprofit group helping people enroll.

At times, more than 125,000 people were simultaneously using HealthCare.gov, straining it beyond its capacity. For long stretches Monday applicants were shuttled to a virtual waiting room where they could leave an email address and be contacted later.

Officials said the site had not crashed but was experiencing very heavy volume. The website, which was receiving 1.5 million visitors a day last week, had recorded about 3 million visits as of 8 p.m. EDT. The call center had received about 1 million calls by that same time.

Supporters of the health care

*See **Insure**, A4*

Another exchange official out

By Nick Budnick
nbudnick@oregonian

Aaron Karjal

A little help from a robot

French inventor Christophe Millot demonstrates the Wall-Ye-France, billed as the world's first robotic vineyard pruner, near Carlton on Monday. Cameras guide its actions, and software remembers every cut from season to season. Millot has another robot that shoos birds and deer away from grapes.

Story in Business, B5

COVER OREGON

```
        // Pause so you can see the output, hit enter to continue
        Console.WriteLine( "Press any key to exit..." );
        Console.ReadLine();
        return;
    }
}
```

COVER OREGON Background

- In 2011 Oregon decided to develop its own state-level health exchange rather than use the Federal government’s healthcare.gov
- Work began on COVER OREGON in 2012, for an October 1, 2013 “go live” date
- Oregon contracted with Oracle to develop the exchange
- The State of Oregon received $300 million in Federal Grant money to develop the site (vs. $100 million planned for healthcare.gov …)
- The exchange was still not working in December 2013, and Oregon reassigned **500 staff** to process **paper** applications
- By April 2014 the exchange was still not working; COVER OREGON was closed, and Oregon adopted healthcare.gov beginning in 2015

COVER OREGON Business Judgment

(Bad Judgment is not Limited to Software Professionals!)

KATU.COM

April 24, 2014

So far, Cover Oregon and OHA have spent two thirds of that money on the exchange, which amounts to $199,199,688

OREGONLIVE
The Oregonian

April 22, 2014

To date, Cover Oregon has signed up more than 63,000 people for private insurance, which generates a per-member per-month fee of $9.38 for the exchange.

– $199,199,688 / 63,000 = **$3,162 per enrollee**
(that's just to access the exchange, **no actual healthcare** included!)

$3,162 per enrollee / **$9.38 per month**, means payoff will require 337 months **(28 years!)**

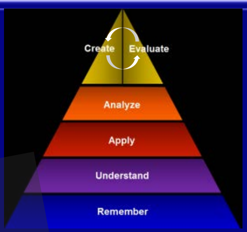
Except that these **people were signed up on paper**, i.e., didn't actually use the exchange!

Reported Problems with COVER OREGON

- ❖ “Code quality is sub-par”
- ❖ “No impact analysis prior to coding”
- ❖ No peer review
- ❖ “Details on software-check-out/check-in and merge processes are lacking”
- ❖ “Build process seems vague and not well defined”
- ❖ “No skilled software development engineering manager”
- ❖ Status reporting “Lacks basic information including number of calendar days and man-days required for project completion”
- ❖ “Poor design”
- ❖ “Even worse code”
- ❖ “The quality of work was atrocious”
- ❖ “They broke every single development best practice that Oracle themselves have defined”
- ❖ “OHA and Cover Oregon lacked the skills, knowledge or ability to be successful”

Source: KATU .com website, April 24, 2014

The Four Factors Model Applied to COVER OREGON



There is Nothing Subtle About What Went Wrong With This Project

Size

- Huge budget
- Huge staff ramp-up
- No aspect of development matched to project size (management, design, construction, CM, test, etc.)

Uncertainty

- Doesn't matter

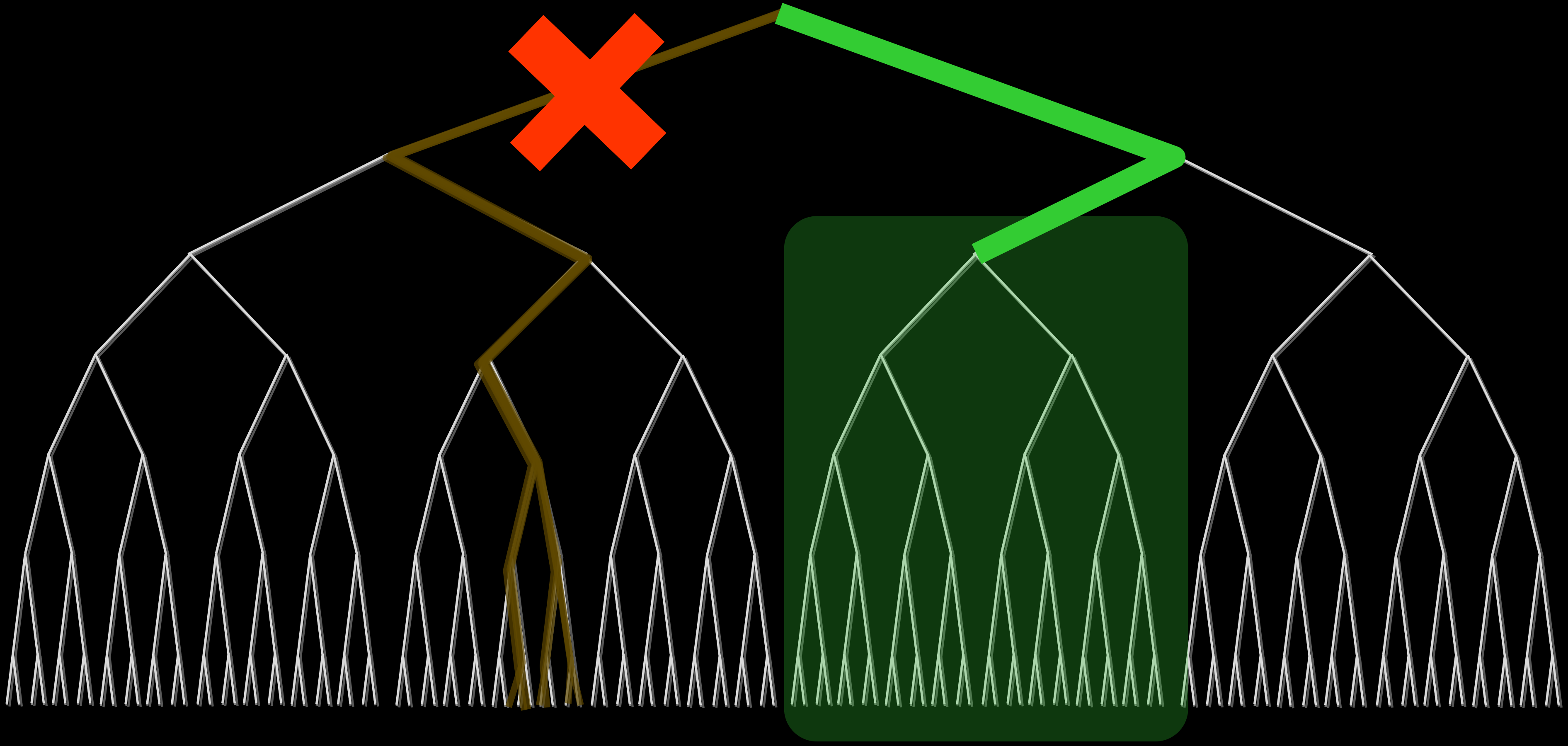
Defects

- Zero meaningful QA practices in evidence

Human Variation

- Massively under skilled

Large-Grain Decisions Were Wrong



“The 1980s are Calling; They Want Their Project Back”

- The problems on this project were so conspicuous that the case study seems almost contrived to make a point—but it is not
- You would think we would have learned these lessons **decades ago**, but this project was still ongoing **less than one year ago**

Aren’t the Problems with this Project Obvious?

- ***Made to Stick*** describes the **Curse of Knowledge**
- I’ve been doing this for a **long time**
- The more time goes by, the more **difficulty** I have knowing **what is obvious** to other people and **what is not**
- The problems with this project seem **obvious to me**
- Yet … this project was allowed to go wrong, by intelligent people, with multiple levels of oversight, to the tune of $**200 million**

Commonalities with Other Case Studies

- The problem was not absence of analysis, not subtle miscalculations, not subtle errors in judgment, but **Gross Errors in Judgment**

- We're asking the wrong question:
  "What went wrong with this project?"

  The right question is,
  "Why did **Anyone—Ever**—think this project would be successful?"

“Train Wreck”

Case Studies

Train Wreck Chronology

The Scene:

In Seattle, a traditional "brick and mortar" parts company (**“The Client”**) decides it wants to take its business online. It does not have any software development capability, so it decides to outsource

It raises $1.7M in investment capital, identifies a high-flying internet company that it would like to work with (**"The Contractor"**), and the project begins.

Train Wreck Chronology

January Contract Negotiations

$1.7M startup capital

4-5 month delivery schedule

January SOW Signed

Contractor bid in 2 phases, with expectation that $1.7M budget for the total project was achievable

Project start of March 18

Contractor would use RUP--The Rational Unified Process

Train Wreck Chronology

March "Inception Phase“

"Inception" would be followed by Elaboration, Construction, and Transition in Phase 2

Deliverables were **Requirements** (via **Use Cases**) and **Architecture**

30 days – planned completion of **Inception Phase** on **April 18**

$400,000

Billing rates range from $150-$700/hour

Nearly all client staff is based in Chicago and spends Monday mornings and Friday afternoons on airplanes

Train Wreck Chronology

April Declared done with "Inception" phase

Initial Bid for **"Elaboration Phase"** (**not including Construction** or **Transition** phases) of **$1.3 million**

This will consume the **client's entire budget**, before getting to Construction

Contract Negotiations begin

Train Wreck Chronology

May

Budget for remainder of project of $1.7M (total of $2.1M) + Client gives up a 15% Equity stake in their company

Plan of ~**50 staff months** of work (in less than 3 calendar months)

Planned live launch on July 11

Short schedule justified because this is an Integration project, not a custom build

May SOW Signed

Inception team staff leaves; Elaboration team staff begins

Began Working on Elaboration

Finished creating **Use Cases**, which amounted to **17 3” 3-ring binders**

Announced 1-week **schedule slip on 5/5**

Announced 3.5-week **schedule slip on 5/26**

Train Wreck Chronology

June

Announced 1-week **schedule slip** on 6/2 (now **out to 7/18**)

Elaboration team staff leaves; **Implementation team** staff begins

Staff turnover exceeds 200% (i.e., 3 people for each job) in less than 6 months

Implementation team found that the primary tool used for integration was very immature, undocumented, and buggy … making the customization and future modifications longer than expected.

Implementation team finds the 17 3-ring binders of **Use Cases not comprehensible**

Implementation team concludes that schedule goals cannot be met with the RUP approach

Team switches from RUP to Extreme Programming

Train Wreck Chronology

July Announced 3-week schedule slip on 7/29 (to 8/11)

Team begins interviewing client about, “What is the most important story you'd like us to work on this week?”

Client responds, “We want everything that’s enumerated in those 17 3-ring binders”

Team **trims many aspects of Extreme Programming** because there isn't enough time to do them.

Train Wreck Chronology

August

An **internal Contractor document** states it was **impossible** to build this system in **3 months**

Contractor presents a change order to Client saying it needs more money to finish the project

Client begins refusing payment of Contractor's invoices

Train Wreck Chronology

September	***September SOW Signed***
	Client agrees to additional budget for project of $700,000 (total of $2.8M)
	Client agrees to pay past invoices
	Contractor agrees to language that states **if Contractor misses its final delivery**, Contractor must **refund ALL fees** for the project (including January and May SOW fees)

Train Wreck Chronology

October Status is fuzzy; client refuses payment based on missed deliveries

November Status is fuzzy; client refuses payment based on missed deliveries

December Contractor sues client, saying it was on track and client owes it fees for past work

Client counter-sues Contractor saying all its prior fees should be refunded due to missed goals

Train Wreck Chronology

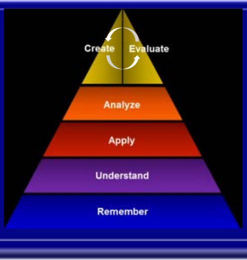
July (year 2)	I get involved as expert witness
September (year 2)	Case settles; client recovers $150,000 (of $2.8 million)
December (year 2)	Client goes out of business
January (year 3)	Contractor acquired by another company for pennies a share (essentially goes out of business)

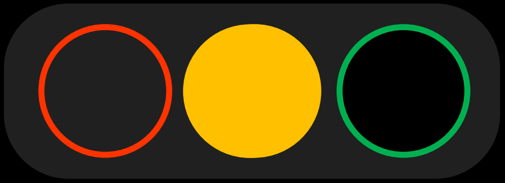
Opposing Expert Witness's Summary

"There were deficiencies in project management, software construction, software design, software configuration management, estimation, software quality assurance, and software testing practices …"

That was from **The Contractor's** expert!

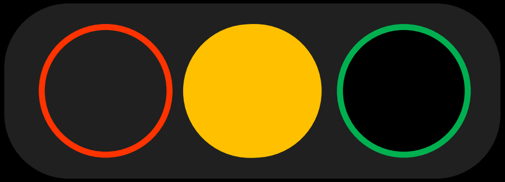
The Four Factors Model Applied to the Train Wreck Project





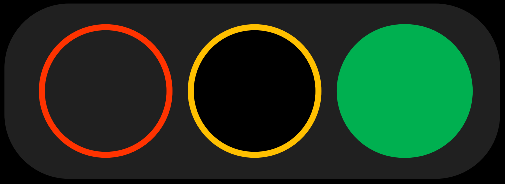
Size

- Not a terribly large project
- Underscoped



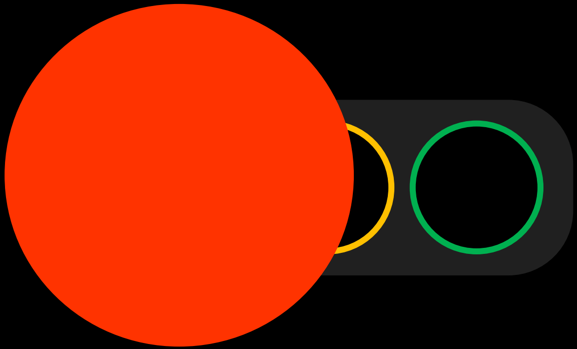
Uncertainty

- There was some technology uncertainty
- All the other uncertainty was introduced by the project team itself



Defects

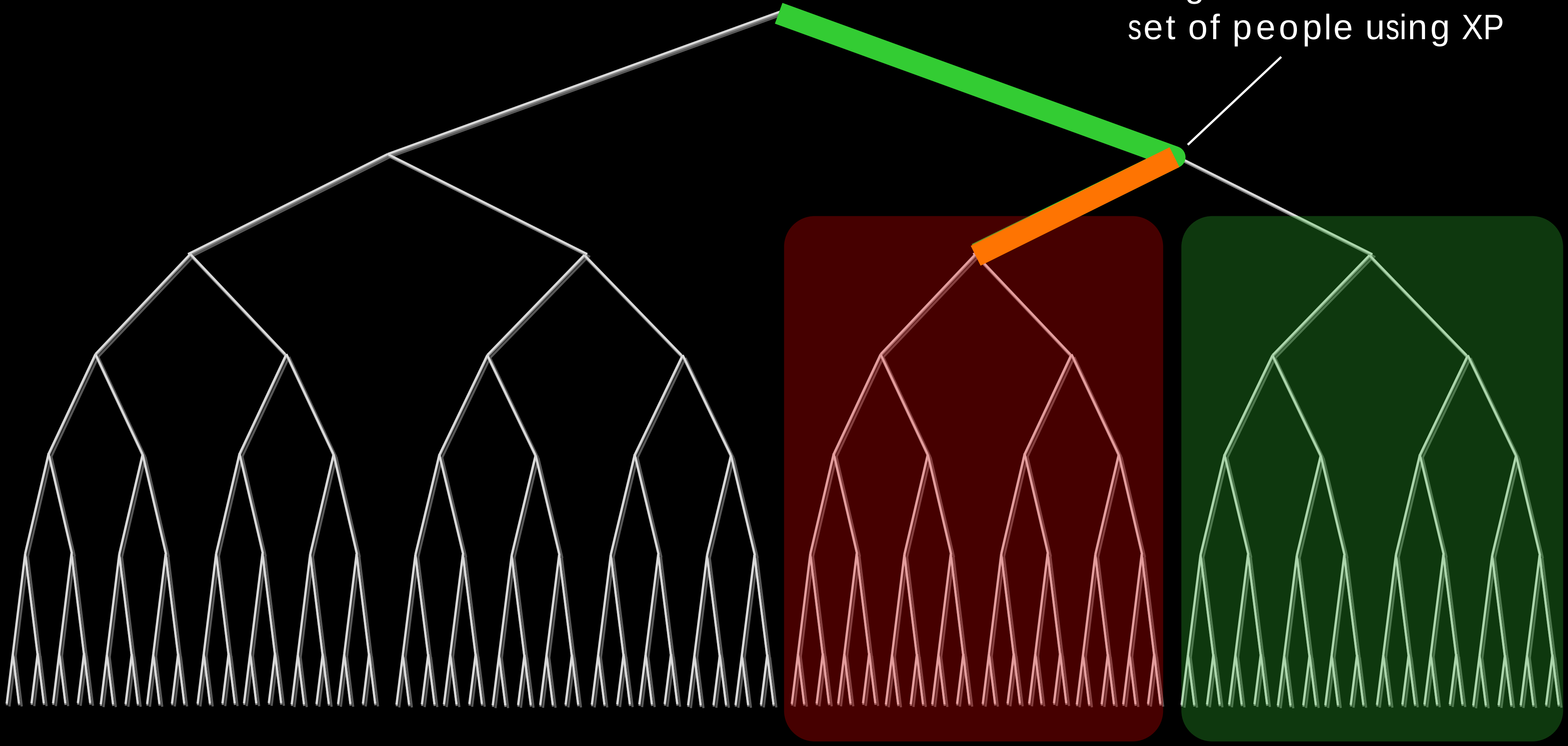
- Pretty good upfront practices with RUP and Use Case analysis



Human Variation

- **Almost incomprehensible** failure to account for human variation in ability to apply RUP vs. XP

I Only Need to Know One Thing About This Project to Predict the Outcome



Decision to switch from one set of people using RUP to a different set of people using XP

Commonalities with Other Case Studies

- Again, there is **nothing subtle** about what went wrong with this project
- As with COVER OREGON and Healthcare.gov, there is plenty of blame to go around
- In cases like this, often **both parties** are at fault
  - I like the legal concept of **Joint and Several Liability**
  - I often find it more useful to adopt the frame of mind, "Assume the project will fail and prove to me that it will work" rather than "Assume it will work and prove that it will fail"

Chrysler C3 Project

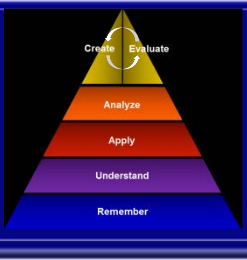
(Original Extreme Programming Project)

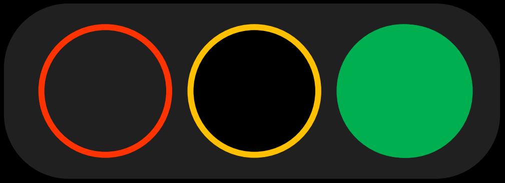
```
class Class1 {

    /// <summary>
    /// The main entry point for the application.
    /// </summary>
    [STAThread]

        // Get properties
        PVSPropertyBag bag = library.GetMediaByFullName( fullName );

        // Print properties to the console
        Console.WriteLine( "Properties of {0}", fullName );
        foreach( PVSProperty prop in bag.Properties ) {
            Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
        }

        // Put the PVSPropertyBag into a more friendly collection class.
        // It's a good idea for you to write a friendlier wrapper class that
        // would allow you to add and remove properties and cast back to
        // the PVSPropertyBag type on the fly.
        ArrayList aProperties = new ArrayList( bag.Properties );

        // Change the "EpisodeDescription" property
        foreach( PVSProperty prop in aProperties ) {
            if( prop.Name == "EpisodeDescription" ) {
                prop.Value = "The boys compete to appear on a talk [illegible] Framework";
            }
        }

        // Create a new PVSPropertyBag with the edited property
        PVSPropertyBag newBag = new PVSPropertyBag();
        newBag.Properties = (PVSProperty[])aProperties.ToArray( typeof( PVSProperty ) );

        // This method will edit the recording
        library.EditMedia( fullName, newBag );

        // Print properties to the console and verify the changes
        Console.WriteLine( "Edited properties of {0}", fullName );
        foreach( PVSProperty prop in bag.Properties ) {
            Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
        }

        // Pause so you can see the output, hit enter to continue
        Console.WriteLine( "Press any key to exit..." );
        Console.ReadLine();
        return;
    }
}
```

Chrysler C3 Project Background (The XP Poster Project)

- Chrysler wanted to replace disparate legacy COBOL payroll systems with one system
- Project did not make much progress from 1993-1995
- In 1996, Kent Beck was hired to build the system; he in turn hired Ron Jeffries
- Kent and Ron implemented pair programming, continuous integration, onsite customer, unit testing, refactoring, YAGNI—all the practices that became Extreme Programming
- Initial release was 2 months late on a 12 month schedule, which the team considered to be "basically on time"
- Progress for the next few years was mixed and characterized by "just one more requirement" syndrome
- Further releases were halted when Daimler bought Chrysler in 2000

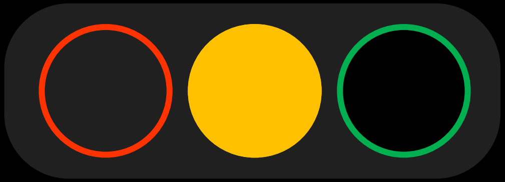
The Four Factors Model Applied to the Chrysler C3 Project





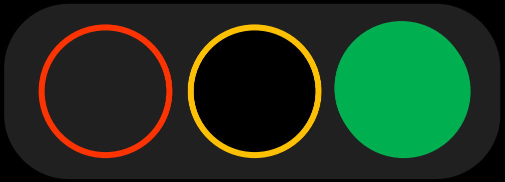
Size

- Small project
- Planned scope pretty close to real scope



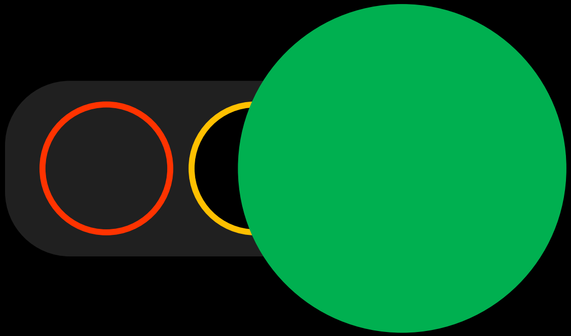
Uncertainty

- Payroll is a well-understood area
- Some uncertainty from the panoply of legacy systems



Defects

- Practices for removing defects were reasonable, and matched to project size
- This is not a high-defect-potential project in the first place



Human Variation

- Kent Beck!
- Ron Jeffries!

Chrysler C3 Project

- Based on the Four Factors model, what surprises me about the Chrysler C3 project?

  NOTHING!

- There is certainly no "XP Secret Sauce" that I would consider significant on the C3 project
- "Why did anyone ever think this project would be successful?"
  - To me, the lesson of the Chrysler C3 project is not about Extreme Programming.
  - The lesson is, "If you pay attention to the needs of the project, and plan and execute accordingly, the project will be successful."

Cheyenne Mountain ATAMS

ATAMS Context

- The US Air Force's Cheyenne Mountain Upgrade project (CMU) was originally scheduled to last 6 years and cost $968M
- Thirteen years later the GAO estimated that CMU was $1 billion over budget and 11 years behind schedule
- The new systems that had been completed were not usable

ATAMS Background

Against this backdrop …

- CMU managers commissioned Kaman Sciences to conduct the ATAMS project
- Goal: **replace** displays on **20 monitors** with **just two** and improve response time
- Project Constraints: Schedule of **one year** and budget of **$2 million**

ATAMS Background

- Kaman Sciences appointed an **experienced project manager**
- Development was conducted by 11-person, **intact development team**
- The team **extensive prototyped** the Ux
- User demands turned a 2-message, 4-display system into a 57-message, 35-display system
  - This was discovered during prototyping
- The team tackled the **riskiest elements first**
- Design reviews caught more than 200 major defects and 500 minor defects at design time at a cost of slightly less than 1 staff hour per defect found

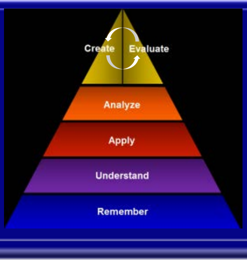
ATAMS Background

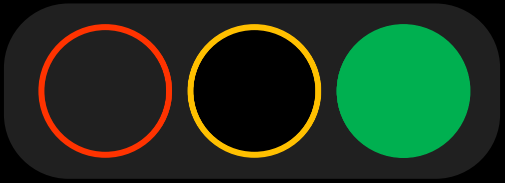
- **Root cause analysis** was performed for each defect found
- **Technical peer reviews** continued throughout the project
- **Active management** was conducted to ensure that peer reviews were performed in a timely way
- Team adopted a standard of **perfecting each component** before moving on to the next component
- Project **status** and tasks status were displayed in a **graphic format** that anyone could understand
- Project management used status information to **seek out project risks** and address them

ATAMS Results

- Delivered **1 month early** on a 12 month schedule
- **Only 2 defects** found within first 16 months of operation

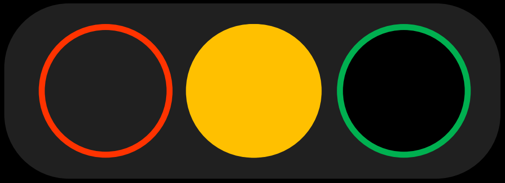
The Four Factors Model Applied to Cheyenne Mountain ATAMS





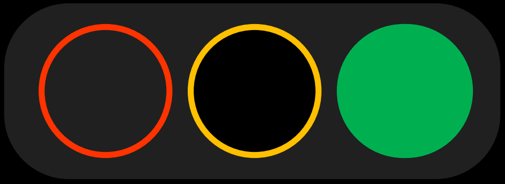
Size

- Small project (11 people)
- Short schedule (1 year)



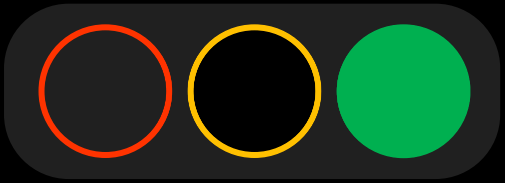
Uncertainty

- Significant requirements changes, but discovered early
- Project actively attacked uncertainty in requirements, quality, and project plans



Defects

- Early requirements defect detection through prototyping
- Thorough reviews
- Focus on maintaining high quality
- High discipline



Human Variation

- Skilled project team
- Skilled management
- Intact team

ATAMS Summary

Compare to commonalities from other projects we’ve seen:

- ❖ “People on the project seem unable to identify even basic dynamics on their own projects, even in hindsight?”
  - ◆ There was an awareness of risk and explicit steps taken to address risks

ATAMS Summary

Compare to commonalities from other projects we've seen:

- "Why did anyone ever think this project would be successful?"
  - Lots of reasons for this project to be successful

ATAMS Summary

Compare to commonalities from other projects we've seen:

- “Problems are not subtleties, but gross errors in judgment”?
  - There were no gross errors in judgment
  - Causes of success in this project seem as conspicuous as causes of failure did on the other projects

Summary

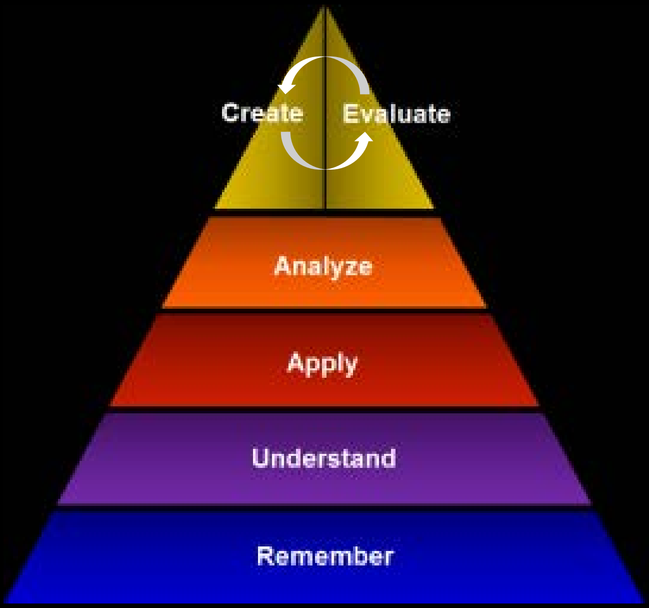
```
class Class1 {

        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main(string[] args) {

                // Logon
                BTVLicenseManager licenseManager = new BTVLicenseManager();

                // [illegible]
                string networkLicense = "00000000-0000-0000-0000-000000000000-00000000";
                string password;

                password = "";
                licenseManager.Logon( networkLicense, password );

                Console.WriteLine( "Logged on." );

                string  fullName = @"C:\Documents and Settings\rkuo.SNAPSTREAM\My Documents\My Videos\South Park-(Freak Strike)-2004-08-17-0.mpg";

                BTVLibrary library = new BTVLibrary();

                // Get properties
                PVSPropertyBag bag = library.GetMediaByFullName( fullName );

                // Print properties to the console
                Console.WriteLine( "Properties of {0}", fullName );
                foreach( PVSProperty prop in bag.Properties ) {
                        Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
                }

                // Put the PVSPropertyBag into a more friendly collection class.
                // It's a good idea for you to write a friendlier wrapper class that
                // would allow you to add and remove properties and cast back to
                // the PVSPropertyBag type on the fly.
                ArrayList aProperties = new ArrayList( bag.Properties );

                // Change the "EpisodeDescription" property
                foreach( PVSProperty prop in aProperties ) {
                        if( prop.Name == "EpisodeDescription" ) {
                                prop.Value = "The boys compete to appear on a talk show. (Edited by Beyond TV Framework)";
                        }
                }

                // Create a new PVSPropertyBag with the edited property
                PVSPropertyBag newBag = new PVSPropertyBag();
                newBag.Properties = (PVSProperty[])aProperties.ToArray( typeof(PVSProperty) );

                // This method will edit the recording
                library.EditMedia( fullName, newBag );

                // Print properties to the console and verify the change
                Console.WriteLine( "Edited properties of {0}", fullName );
                foreach( PVSProperty prop in bag.Properties ) {
                        Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
                }

                // Pause so you can see the output, hit enter to continue
                Console.WriteLine( "Press any key to exit..." );
                Console.ReadLine();
                return;
        }
}
}
```

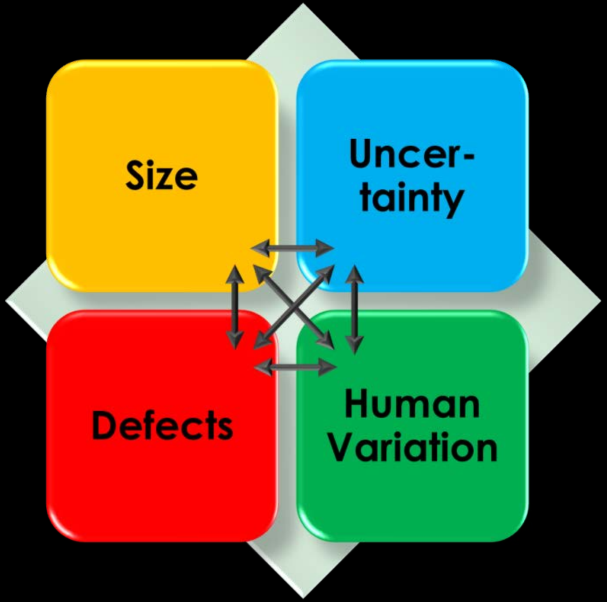
Summary

- Most of what I have described today seems **obvious to me** (the **Curse of Knowledge**)
- However, one common theme in the failed projects is that **basic project dynamics** were **not obvious** to the people involved in these projects, **even highly intelligent people**, often **even in hindsight**
- How can people who are so smart make such bad decisions?
- Software professionals tend to be very strong in Analysis, so deficiency in **Analysis** does **not seem to be the problem**

Summary

- **Deficiency in Judgment**, even **Gross Errors in Judgment** are common in software
- A focus on **Developing Judgment** in software professionals is important, perhaps more important than in professions that do not select so strongly for Analysis skills





Construx®

Software Development Best Practices

Construx Software is committed to helping individuals and organizations improve their software development practices. For information about our training and consulting services, contact

stevemcc@construx.com
+1(425) 636-0100

Construx

10900 NE 8th Street, Suite 1350
Bellevue, WA 98004
+1 (866) 296-6300
www.construx.com

END

```
class Class1 {

    /// <summary>
    /// The main entry point for the application.
    /// </summary>
    [STAThread]
    static void Main(string[] args) {

        // Logon
        BTVLicenseManager licenseManager = new BTVLicenseManager();

        // put your valid license here
        string  networkLicense = "00000000-0000-0000-0000-000000000000-00000000";
        st[illegible] password;

        pa[illegible]
        li[illegible]Logon[illegible]kLicens[illegible]sswo[illegible]

        Co[illegible]e.WriteLine( "[illegible]" );

        st[illegible]fullName = @[illegible]ts a[illegible]ttin[illegible]uo.SNAPSTR[illegible]y Documents\My Videos\South Park-(Freak Strike)-2004-08-17-0.mpg";

        BT[illegible]ry = [illegible] BT[illegible]ry[illegible]

        // [illegible] properties
        PV[illegible]pertyBag bag = [illegible]ary.G[illegible]ullN[illegible]fullName[illegible]

        // [illegible]e cons[illegible]
        Co[illegible]( "[illegible]rties o[illegible]full[illegible]
        fo[illegible]rty[illegible] in bag.[illegible]ies[illegible]
            [illegible]Lin[illegible]roperty: [illegible]1}", [illegible] prop.Value );
        }

        // Put the PVSPropertyBag into a more friendly collection class.
        // It's a good idea for you to write a friendlier wrapper class that
        // would allow you to add and remove properties and cast back to
        // the PVSPropertyBag type on the fly.
        ArrayList aProperties = new ArrayList( bag.Properties );

        // Change the "EpisodeDescription" property
        foreach( PVSProperty prop in aProperties ) {
            if( prop.Name == "EpisodeDescription" ) {
                prop.Value = "The boys compete to appear on a talk show. (Edited by Beyond TV Framework)";
            }
        }

        // Create a new PVSPropertyBag with the edited property
        PVSPropertyBag newBag = new PVSPropertyBag();
        newBag.Properties = (PVSProperty[])aProperties.ToArray( typeof(PVSProperty) );

        // This method will edit the recording
        library.EditMedia( fullName, newBag );

        // Print properties to the console and verify the change
        Console.WriteLine( "Edited properties of {0}", fullName );
        foreach( PVSProperty prop in bag.Properties ) {
            Console.WriteLine( "Property: {0}, {1}", prop.Name, prop.Value );
        }

        // Pause so you can see the output, hit enter to continue
        Console.WriteLine( "Press any key to exit..." );
        Console.ReadLine();
        return;
    }
  }
}
```

ACM: The Learning Continues…

- Questions about this webcast? learning@acm.org
- ACM Learning Webinars (on-demand archive): http://learning.acm.org/webinar
- ACM Learning Center: http://learning.acm.org
- ACM SIGSOFT: http://www.sigsoft.org/
- ACM Queue: http://queue.acm.org/