Industrial Strength Software Measurement

David Weiss*
Lanh & Oanh Nguyen Professor of Software Engineering
Iowa State University
weiss@iastate.edu

* With participation by Jim Kirby, Robyn Lutz, and data/analysis from Audris Mockus, Randy Hackbarth, John Palframan, et al.
Topics

• What and How to Measure
  – What do they do in other industries?
  – Goal Question Metric Paradigm
  – Change Measurement
  – Goals and Context
  – Practical Issues: How to get answers

• Some Examples
  – Predictability
  – Customer Satisfaction: Interval Quality

• Evidence-based improvement: What could we learn?
What do they do in other industries?
U.S. corn area and yield

USDA, Economic Research Service.
Estimated global meat production

Million metric tons

Calendar year

USDA, Economic Research Service.
Top Ten Vehicle Assembly Plants

Hours Per Vehicle

<table>
<thead>
<tr>
<th>Plant</th>
<th>Models</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toledo South</td>
<td>Wrangler, Wrangler Extended</td>
<td>13.57</td>
</tr>
<tr>
<td>Oshawa #1</td>
<td>Impala, Monte Carlo</td>
<td>15.18</td>
</tr>
<tr>
<td>Oshawa #2</td>
<td>Grand Prix, LaCrosse</td>
<td>16.17</td>
</tr>
<tr>
<td>Belvidere</td>
<td>Caliber, Compass, Patriot</td>
<td>17.09</td>
</tr>
<tr>
<td>CAMI</td>
<td>Equinox, Torrent, XL-7</td>
<td>17.59</td>
</tr>
<tr>
<td>Kansas City #1</td>
<td>Escape, Mariner, Tribute (also HEV variants)</td>
<td>17.72</td>
</tr>
<tr>
<td>Lordstown</td>
<td>Cobalt, G5</td>
<td>18.12</td>
</tr>
<tr>
<td>Jefferson North</td>
<td>Commander, Grand Cherokee</td>
<td>18.68</td>
</tr>
<tr>
<td>Brampton</td>
<td>300, Charger, Magnum</td>
<td>18.78</td>
</tr>
<tr>
<td>NUMMI Car</td>
<td>Corolla, Vibe</td>
<td>18.79</td>
</tr>
</tbody>
</table>
North American Pre-Tax Profit Per Vehicle

Chrysler – data through August 4, 2007
Ford/GM – data based on calendar year
Honda, Nissan & Toyota – data based on fiscal year (4/1/07 – 3/31/08)
Currency Conversions: 115 Yen/1.37 Euro
## Table 1. Productivity and related indexes for the semiconductor industry, 1972–1986

<table>
<thead>
<tr>
<th>Year</th>
<th>Output per employee hour</th>
<th>Output</th>
<th>All employee hours</th>
<th>Employees</th>
</tr>
</thead>
<tbody>
<tr>
<td>1972</td>
<td>46.3</td>
<td>36.4</td>
<td>78.7</td>
<td>78.1</td>
</tr>
<tr>
<td>1973</td>
<td>53.6</td>
<td>51.0</td>
<td>95.2</td>
<td>94.8</td>
</tr>
<tr>
<td>1974</td>
<td>64.1</td>
<td>63.5</td>
<td>99.0</td>
<td>100.5</td>
</tr>
<tr>
<td>1975</td>
<td>56.0</td>
<td>45.8</td>
<td>81.8</td>
<td>82.5</td>
</tr>
<tr>
<td>1976</td>
<td>82.5</td>
<td>72.4</td>
<td>87.8</td>
<td>88.0</td>
</tr>
<tr>
<td>1977</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>1978</td>
<td>120.5</td>
<td>139.0</td>
<td>115.4</td>
<td>114.5</td>
</tr>
<tr>
<td>1979</td>
<td>138.1</td>
<td>190.4</td>
<td>137.9</td>
<td>136.3</td>
</tr>
<tr>
<td>1980</td>
<td>149.4</td>
<td>226.1</td>
<td>151.3</td>
<td>151.5</td>
</tr>
<tr>
<td>1981</td>
<td>171.6</td>
<td>260.5</td>
<td>151.8</td>
<td>151.7</td>
</tr>
<tr>
<td>1982</td>
<td>197.9</td>
<td>301.2</td>
<td>152.2</td>
<td>153.0</td>
</tr>
<tr>
<td>1983</td>
<td>211.5</td>
<td>339.9</td>
<td>160.7</td>
<td>159.3</td>
</tr>
<tr>
<td>1984</td>
<td>229.2</td>
<td>432.5</td>
<td>188.7</td>
<td>185.6</td>
</tr>
<tr>
<td>1985</td>
<td>206.1</td>
<td>392.3</td>
<td>190.3</td>
<td>189.4</td>
</tr>
<tr>
<td>1986</td>
<td>218.4</td>
<td>399.4</td>
<td>182.9</td>
<td>181.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Average annual rates of change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1972–81</td>
</tr>
<tr>
<td>16.6</td>
</tr>
<tr>
<td>4.1</td>
</tr>
<tr>
<td>1981–86</td>
</tr>
<tr>
<td>25.4</td>
</tr>
<tr>
<td>9.5</td>
</tr>
</tbody>
</table>

### Productivity Gains In The Semiconductor Industry
What Would We Like to Know?

• Depends on goals
  – Use goal-question-metric method
• Define goals
  - Typical goals
    • Improve quality
    • Reduce time-to-market
    • Reduce cost by improving productivity
• Identify questions needed to assess progress towards goals
• Define metrics needed to answer questions
Measurement Approach: GQM

• Identify goals of software development process
  – Example: Produce more new features, fewer defects with fewer, more distributed, resources.

• Propose questions whose answers establish progress towards goals
  – Example: What is the ratio of new features to bug fixes by product? By site?

• Define measures that can be used to answer questions and that can be practically obtained for the software project
  – Example: Ratio of new feature Modification Requests (MRs) to bug fix MRs by product and site, normalized.

• Validate measures internally and externally
  – Example: remove tool generated artifacts and ensure the measure represents the phenomena it is intended to measure

• Establish infrastructure for data collection and analysis
  – Dashboards
  – Automated data collection and analysis
Software Changes: A Fabric of Measurement

• MR = Modification Request
  – For every change
    • Why was it made?
    • Who requested it?
    • Who made the change?
    • What was changed?
    • When was it changed?
    …

• States of an MR
  – Created (Developer, Tester, Support)
  – Assigned (MR Review Board)
  – Submitted (Developer)
  – Verified (Tester)
  – Completed (MR Review Board)
  – Accepted
An Example: Assessment At Avaya

- Who does it?
- How is it done?
- What are the results?

Hackbarth, Mockus, Palframan, Weiss; Assessing the state of software in a large enterprise, JESE, Nov 2009
Avaya Resource Center for Software Technology (ARC)

Business Value
- Improve State of the Practice in Avaya Software Development and Know It
  - Improved time to market, and software quality and decreased software cost

Approach
- Assess the current state of software in Avaya, identifying common problems, create action plans, create dashboards and report on status to Avaya executives.
- Investigate and Introduce new software technology and adapt industry proven techniques to improve Avaya's software production processes
- Help build a community of software competence in Avaya
- Continually improve our approaches of partnering with Avaya organizations

Key Avaya Software Trends

Vision: Make Software a Strategic Weapon for Avaya.
- Software is critical to Avaya’s products and to keeping its customers happy and expanding its market place.
- The Avaya code base tripled between 2000 and 2006 (see graph)

Selected Deliverables and Milestones
- Tailored software improvement programs
- Iterative Development Effectiveness in Avaya
- Predictability Report Update
- Cross-division arch reviews (define/participate)
- Test Forum
- Report on State of Avaya Software

Contact

Dashboard Sample – Avaya R&D is Driving Down Customer Affecting Problems
One Year Trends
All Critical Areas

Legend:
- No change in 2007
- Improving in 2007
- Worsening in 2007

(*) There are separate arrows for Test Automation and Testing
Some Avaya Goals (1)

- Significantly improve predictability
  - Is predictability improving?
  - What fraction of projects are on time?
  - What are the factors associated with late projects?
  - Are there differences in productivity, quality among sites?
- Significantly improve quality
  - Is quality improving?
  - What is the customers’ perceptions of software quality?
  - What is the in-process quality?
- Rapidly produce new products (days and weeks instead of months and years)
Some Avaya Goals (2)

- Keep production within limits of resources, which are becoming more distributed
  - How distributed are resources? What’s the trend?
  - Are there differences in productivity, quality among sites?
- Make globally distributed development (independent component development at different sites) an advantage
  - Are there differences in productivity, quality among sites?
- Introduce new software development processes
  - Agile development using automated test tools
  - What processes do we use?
- How well do projects using a process meet their deadlines?
  - Is there a difference in quality when different processes are used?
Some Avaya Goals (1)

- **Significantly improve predictability**
  - Is predictability improving?
  - What fraction of projects are on time?
  - What are the factors associated with late projects?

- **Significantly improve quality**
  - Is quality improving?
  - What is the customers’ perceptions of software quality?
  - What is the in-process quality?

- **Are there differences in productivity, quality among sites?**

- **Rapidly produce new products (days and weeks instead of months and years)**
  - Use a modular, family architecture
  - Take advantage of commonalities to compose and generate rather than hand code

- **Make production predictable**
  - Continually predict, trial, and leverage expected future needs
  - Develop infrastructure for composing products from modules
Predictability
Predicting Software Development (50 sampled projects)
Distributed development, innovation, new features, legacy adaptation all contribute to delays

**Average additional time to complete than committed at gate 1**

- Single Site
- Multiple Site (Single Product Group)
- Multiple Product Group
- External Partnership

**Average additional time to complete than committed at gate 1**

- Incremental release with small feature set, Bug Fixes and Quality enhancements
- Many new features; some invention.
- Adaptation of legacy system to new architecture
- High degree of invention required
Some Avaya Goals (1)

- Significantly improve predictability
  - Is predictability improving?
  - What fraction of projects are on time?
  - What are the factors associated with late projects?

- Significantly improve quality
  - Is quality improving?
  - What is the customers’ perceptions of software quality?
  - What is the in-process quality?
  - Are there differences in productivity, quality among sites?

- Rapidly produce new products (days and weeks instead of months and years)
  - Use a modular, family architecture
  - Take advantage of commonalities to compose and generate rather than hand code
  - Make production predictable
  - Continually predict, trial, and leverage expected future needs
  - Develop infrastructure for composing products from modules
Assessing Quality
Interval Quality

• Probability that a customer observes a failure within one, three, and six months after installation

• Drawback
  – does not account for the proximity to launch

• Significant differences are marked with *, **, and ***

• Priorities changed from time-to-market to quality

![Bar chart showing probability distribution over releases and months]
Interval Quality and Defect Density

- **X-axis:**
  - Releases

- **Y-axis:**
  - Defect ratio, IQ

- **Features:**
  - negative correlation
  - major releases look better in terms of defect density
ASSESSING THE ORGANIZATION
Multi-Year Trends
Several Critical Areas

Effectiveness

Low

High

Usage

Low

High

Test Automation

Customer-Focused Development

Automated Build Management

Predictability
One Year Trends
All Critical Areas

(*) There are separate arrows for Test Automation and Testing

Legend:
- No change in 2007
- Improving in 2007
- Worsening in 2007
Summary

• Use an evidence-based process to assess your progress
  – Identify your goals
  – Find a good set of questions whose answers help evaluate progress in meeting goals
  – Collect and analyze data that answers the questions
  – Measure progress regularly
Why Do Evidence-Based Assessment for Software?

• Knowing what the key questions are, and answering them, drives improvement

• Similar questions have been answered, and the answers used to drive rapid, sustained progress, in fields as diverse as agriculture, genetics, automotive engineering, particle physics, health care, and semiconductor manufacturing. Software engineering should be no different.
Evidence-Based Industrial Assessment Is Common

• Need consistent, standardized data to assess progress
  – Evidence
    • Of state of the practice in an organization (industry)
    • Of progress towards achieving goals
    • Of the value of different methods and technologies

• Need evidence to assess needs and make predictions
  – How many items can we produce next year?
  – How many items will we sell next year?
  – What is our quality?
  – How effective is (crop rotation, robotic manufacturing, GPS navigation, …)?

• Evidence frequently leads to rapid, major advances
  – Agriculture
    • 1900: Nearly half the workforce engaged in agriculture, 40% of income went to food
    • 1930: 20% of workforce engaged in agriculture, 20% of income went to food
    (Gawande, The New Yorker, December, 2009)
20-Year Lag Time Between Smoking and Lung Cancer

Source: NIH
What About Software?

• Few examples of large-scale data collection
• Little standardization of data
• Widely-varying industrial settings
• Avionics, Automotive, Medical, Energy, Defense, Telecommunications, Entertainment
• Evidence is often anecdotal or based on small-scale trials
• We need to start collecting evidence for software production
  – Production = Creation + Sustainment + Evolution
Proposition

• We could significantly improve software production if we had standardized data across the industry
  – Would enable better understanding of software development processes and technology
  – Routinely done in many industries
  – We need to start now

• Collecting and analyzing the data we need is difficult, but possible and worthwhile
  – Data collection and analysis will get easier over time
Some Interesting Questions (1)

• How much software can a developer produce in a year?

• How many changes per year can a software system sustain before it becomes unmaintainable?
  – What is the impact of generational change on sustainability?

• What is the most crucial knowledge needed to sustain a software system over its expected lifetime?

• How much software critical to the organization will we produce next year? Can we produce next year?

• What human resources are needed to develop and sustain the software critical to the organization?
Some Interesting Questions (2)

• How long does it take to make a change to a software system? What does it cost?
  – How does this vary by domain?

• How many disasters per year result from software defects?

• What process or technology changes have the greatest impact on software productivity? On software quality? What is that impact?

• What factors dominate the production process as the scale of software increases?
  – Size
  – Complexity
  – Length of expected lifetime
Summary

• Why measure?
  – Estimate parameters important to business
    • Customer satisfaction, predictability, time and resources needed to create products
  – Evaluate progress on particular projects
    • When will it be ready? How many architects, developers, testers will we need?
  – Estimate capabilities and needs to understand areas for improvement
    • What problems do we need to solve to improve?
    • What is the impact of introducing new technology, methods?
• Personal, Business, Country, World

• What are characteristics of industrial measurement?
  • Change data as a key information source
  • Automatibility, nonintrusiveness
"When you can measure what you are speaking about, and express it in numbers, you know something about it, but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind: it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science."

William Thomson, Lord Kelvin