Designing Software Platforms for Innovation and Profitability

results from 2 pilot projects
- driving design quality
- managerial decision-making

Martin Jouvenot
Software Architect
Amadeus
(& MIT SDM '17)

Dan Sturtevant
CEO
Silverthread, Inc.
(& MIT SDM Alum)

Rashesh Jethi
Head of R&D - Americas
Amadeus
1. Introducing Amadeus
2. Challenges faced by the software enterprise
3. Introducing Silverthread, Inc.
4. Case 1: Driving software platform design quality
5. Case 2: Analytics for managerial decision-making
6. Lessons for your company
Among the world’s top 10 software companies
Forbes 2016 global rankings

14,000 people
195 countries
4.47B 2016 revenue
Powering the biggest names in travel in North America...
...and Globally!

- 709 airlines
- 580,000 hospitality properties
- 110 airport operators
- 90 rail operators
- 100 ground handlers
- 233 tour operators
- 43 car rental companies
- 90,000 travel agencies and corporations, online and worldwide
- 50 cruise and ferry lines
- 16 Insurance provider groups

© 2017 Amadeus North America, Inc.
Research & Development

Leading 20 R&D centers

The travel technology industry

€ 4B+

Invested since 2004
Maintaining Industry Leadership

- Improved Agility
- Cloud-enabled Platforms
- Data-driven Technology Investments
1. Introducing Amadeus
2. Challenges faced by the software enterprise
3. Introducing Silverthread, Inc.
4. Case 1: Driving software platform design quality
5. Case 2: Analytics for managerial decision-making
6. Lessons for your company
A large portfolio of services

5000+ services

2000+ components

145K queries/second
Impacting a lot of people

- 595M bookings in 2016
- 2M bookings every day
- 1.4B passengers boarded in 2016
- ~2,600 passengers boarded every minute
Large and complex software codebases

“Writing code is like writing poetry: every word, each placement counts. Except that software is harder, because digital poems can have millions of lines which are all somehow interconnected… So far, nobody has found a silver bullet to kill the beast of complexity.”

- Stuart Feldman, IBM Institute for Advanced Commerce (2001)

Useful Metaphor: Software Development = Community Poetry Writing at Massive Scale!
Code lives forever and scales exponentially

Google’s 25,000 Engineers commit 15 million LOC/day - Equivalent to Linux; every day

Software systems rarely die; we build on legacy code. Hence today’s designers inherit past design decisions

Source: https://www.youtube.com/watch?v=W71BTkUbdqE
We deal with multiple, often competing business goals

- Enhance existing solutions
- Build new products
- Sign new customers
- Improve deadlines
- Maximize growth

while also

Investing in the platform
Engineers have little visibility into the design structure of code.

Margaret H. Hamilton
And the scale of that challenge continues to multiply

Apollo 11: 65kLOC
Launched July 16, 1969

Amadeus Today: ~100 Million LOC

\[ \times \]

1500

\[ = \]

Stack of printed paper would be 1.5 miles or 2.4 KM high
As a discipline, software development is very challenging

6.4% of large projects successful

- 41.4% failures
  - Abandoned
  - Started again from scratch
- 52% challenged
  - Budget overrun
  - Schedule overrun
  - Bad functionality

The Standish Group,
with a database of 50,000 development projects
http://www.computerworld.com/s/article/9243396/
Healthcare.gov_website_didn_t_have_a_chance_in_hell
October 24, 2013, a jury ruled against Toyota and found that unintended acceleration could have been caused due to deficiencies in the drive-by-wire throttle system or Electronic Throttle Control System (ETCS).

2013 Cambridge University Study Found Software Bugs Cost (Global) Economy $312 Billion Per Year (just cost to fix defects)
Given all of these challenges, our talk today is about 2 pilot projects led by Amadeus & Silverthread aiming to:

1. Help technology leaders confidently evolve the software platform
2. Help business leaders optimize business outcomes with analytics

Help both communicate and collaborate more effectively
1. Introducing Amadeus
2. Challenges faced by the software enterprise
3. Introducing Silverthread, Inc.
4. Case 1: Driving software platform design quality
5. Case 2: Analytics for managerial decision-making
6. Lessons for your company

© 2017 Amadeus North America, Inc.
Who is Silverthread?

Dan Sturtevant  Michael Davies
Founder and CEO  Founder & Chairman
MIT tech strategy faculty

Alan MacCormack  Carliss Baldwin
Founder & Board Member  Founder & Board Member
Harvard business faculty  Harvard finance faculty

Sean Gilliland  Walker Royce
Director of SW Dev  Chief Software Economist
Silverthread drives results in software products and portfolios

Our Customers

| Software Enterprise | Government & Defense | Due Diligence |

Areas of Focus

- Design Quality Measures
- Code Quality Measures
- Software Economics Framework

Key Questions

Diagnose
- How complex is my system?
- Where are potential tumors?
- Where will we struggle with cloud transition?

Assess
- Is architectural integrity getting better or worse?
- How does we compare with similar systems?
- Where does the as-coded system diverge from design intent?

Recommend
- Where should we invest in highest priority improvements?
- Should we refactor, redesign or continue with maintenance?
- How can we capture more persuasive business cases?
Silverthread: More honest measurement and steering

**CodeMRI® Reports & Tools**

**Descriptive analytics ➞ for architects**
- Visuals of potential tumors
- Benchmarking
- Correlation of quality hotspots and defect trends
- Correlation between as-coded baseline and intended design

**Predictive analytics ➞ for leaders**
- Maintainability, security, quality, productivity improvements
- Quantify technical debt
- Predict estimated ROI of targeted refactoring

**Empirical platform ➞ for enterprises**
- Always-on DevOps CodeMRI®
- Context relevant benchmarking
- Refactoring insight
- Preventive health for development

**CodeEKG™ and Zoo**

(Under development)

**Professional Services**

**Change catalysts ➞ for projects**
- Ramp-up training
- Customized reports or insights
- Custom integration with other tools
- Expert testimony for corroboration
- Executive persuasion

© 2017 Silverthread, Inc.
Two pilot projects led by Amadeus & Silverthread

1. Gain insights about the software
   Allow software architects to understand the design structure of Amadeus’ enterprise software platform, drive out architectural issues, and be agile at large scale

2. Get the biggest bang for the buck
   Help senior leadership make data driven and financially rational choices about investments in continuous improvement efforts to optimize business outcomes.
1. Introducing Amadeus
2. Challenges faced by the software enterprise
3. Introducing Silverthread, Inc.
4. Case 1: Driving software platform design quality
5. Case 2: Analytics for managerial decision-making
6. Lessons for your company
Design quality: Properties that help manage complexity

Well architected system

- Developers understand the code, system adaptable

Poorly architected system

- Development team confused and frustrated, system inflexible & brittle

Platforms

- Locality of change
- Abstraction

Modularity

- Locality of change
- Fan-in, Fan-out
- Core size

Reuse

- Locality of change
- Reuse
- Replication

Hierarchies

- Locality of change
- Cyclicality
- Propagation cost

Comp Quality

- Defect density
- Coverage tools
- Unit test scans

How do we quantify?

© 2017 Silverthread, Inc.
Illustrating design quality concepts with simple examples

In Hardware

Platforms & Hierarchy

Modularity & Reuse

In Software

Network Diagrams?

Visually appealing
Hard to Interpret
How else can we visualize software architecture?

Direct Dependencies

Indirect Dependencies

Traditional network view

Matrix notation

© 2017 Silverthread, Inc.
What an ‘ideal’ software system looks like

Design Structure Matrices (DSMs) are network representations of complex systems.
What does it look like when design quality breaks down?

Modular System

Integral System

© 2017 Silverthread, Inc.
Design intent vs the coded reality

Declared relationships extracted from the build files

Actual relationships extracted from inter-file dependencies

Hierarchy

Reuse

Modularity
Structural analysis to spot irregularities and risks

- Unexpected dependencies
  - Hidden complexity
- Cyclic dependencies
  - Tight coupling
  - Unintended consequences
- Tightly coupled core
  - Area of high complexity
  - Difficult to debug
1. Introducing Amadeus
2. Challenges faced by the software enterprise
3. Introducing Silverthread, Inc.
4. Case 1: Driving software platform design quality
5. Case 2: Analytics for managerial decision-making
6. Lessons for your company
Studies: Quality has a big impact on productivity & quality

Parts with higher design quality

- Developer productivity: 80KLOC per year
- Feature work: 80%
- Defect work: 20%

Parts with lower design quality

- Developer productivity: 30KLOC per year
- Feature work: 30%
- Defect work: 70%
Studies: Quality has a big impact on **safety** and **security**

**System safety**

Fortune 100 engineering conglomerate

Design quality degradation increased defects found after a safety-critical system went live

<table>
<thead>
<tr>
<th>% of fielded system with critical bugs</th>
<th>High quality design</th>
<th>Design quality degraded</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.63%</td>
<td>13.00%</td>
<td></td>
</tr>
</tbody>
</table>

| Dollars spent fixing critical defects per LOC | 12 cents | 81 cents |

**Security threat**

Fortune 50 consumer software firm

Design quality degradation responsible for security vulnerabilities and higher maintenance costs

In code with a measurably better architecture:
- Fewer vulnerabilities & defects found
- 10% higher developer productivity during patch process
- 14% less time to release patches
- 25% fewer incomplete or incorrect fixes
Studies: Quality has a big impact on **agility** and **revenue**

Software platform revisions at a Fortune 500 firm over a 9 year period

- **Refactoring underway**
- **Refactoring complete**
- **Rate of new product development accelerates dramatically post-refactoring**

Number of revenue generating products

© 2017 Silverthread, Inc.
<table>
<thead>
<tr>
<th>Engineer Suggests</th>
<th>Market Demand</th>
<th>ROI</th>
<th>CFO Reaction</th>
</tr>
</thead>
</table>
| New Features     | • Existing customer will buy more  
                  • New customers created  
                  • Product differentiation | $     | Ecstatic              |
| Bug Fixes        | • Existing customer screaming  
                  • New customers avoiding  
                  • Reputation hit          | $     | Required, unfortunately |
| Internal Quality Improvement, Refactoring Initiatives | None | !                    | Not convinced         |
Mechanics: Statistically linking quality & business outcomes

**Product Attributes**
- Source code
- Continuous integration

**Process Performance**
- Version control
- Feature & bug tracking

**Data Sources**
- Source code
- Continuous integration

**Raw Metrics**
- Code complexity metrics
- Design complexity metrics
- Language
- File attributes
- Design structure
- Unit test coverage
- System test coverage
- Test failure rate
- Change volume
- Change timestamp
- Change locality
- File age
- Developer ID
- Bug vs Feature?
- Criticality
- Time to close issue
- Developer ID
- Development effort

**Stats Models**
- Code quality
- Design quality
- Test quality
- Defect density
- Dev productivity
- Team morale
- Change locality
- On-time delivery
- Cost of change
Example regression model linking quality and developer productivity

Predicting LOC produced by a developer to implement enhancements for one release. (Negative binomial panel data model)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Model 1: developer attributes</th>
<th>Model 2: type of work</th>
<th>Model 3: cyclomatic complexity</th>
<th>Model 4: all controls</th>
<th>Model 5: architectural complexity</th>
<th>Model 6: combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lines for bug fixes</td>
<td>-0.000071</td>
<td>-0.000068</td>
<td>-0.000060</td>
<td>-0.000067</td>
<td>-0.000077</td>
<td>-0.000078</td>
</tr>
<tr>
<td>Log(years employed)</td>
<td>0.279600</td>
<td></td>
<td>0.492500</td>
<td></td>
<td>0.483700</td>
<td></td>
</tr>
<tr>
<td>Is manager?</td>
<td>-0.283000</td>
<td></td>
<td>-0.251600</td>
<td></td>
<td>-0.292900</td>
<td></td>
</tr>
<tr>
<td>Pct lines in new files</td>
<td>1.801000 ***</td>
<td>1.699000 ***</td>
<td>1.714000 ***</td>
<td>1.714000 ***</td>
<td>1.714000 ***</td>
<td></td>
</tr>
<tr>
<td>Pct lines high cyclomatic</td>
<td>-1.166011 ***</td>
<td>-0.648300</td>
<td></td>
<td></td>
<td>-0.613000</td>
<td></td>
</tr>
<tr>
<td>Pct lines in core</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-0.610943</td>
<td>-0.618600 *</td>
</tr>
<tr>
<td>Residual Deviance</td>
<td>560.77</td>
<td>558.46</td>
<td>560.60</td>
<td>558.32</td>
<td>560.71</td>
<td>558.13</td>
</tr>
<tr>
<td>Degrees of Freedom</td>
<td>290.00</td>
<td>291.00</td>
<td>291.00</td>
<td>288.00</td>
<td>291.00</td>
<td>287.00</td>
</tr>
<tr>
<td>AIC</td>
<td>8170.66</td>
<td>8135.14</td>
<td>8162.14</td>
<td>8136.78</td>
<td>8166.87</td>
<td>8135.75</td>
</tr>
<tr>
<td>Theta</td>
<td>0.85</td>
<td>0.90</td>
<td>0.86</td>
<td>0.91</td>
<td>0.85</td>
<td>0.92</td>
</tr>
<tr>
<td>Std-err</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>2 x log-lik</td>
<td>-7792.66</td>
<td>-7759.14</td>
<td>-7786.14</td>
<td>-7754.78</td>
<td>-7790.87</td>
<td>-7751.75</td>
</tr>
</tbody>
</table>

N = 478 developer/releases

Dummy variables for each of 8 releases omitted. Dummy variables for each of 178 developers omitted.

Significance codes: .<0.1, *<0.05, **<0.01, ***<0.001
How to win over your CFO in 6 easy steps!

Step 1: A hypothetical proposal

☐ Come up with a hypothetical proposal
  • “I think we should increase test coverage in our codebase. It will allow us to catch bugs earlier and prevent their introduction. We have some source files with no automated unit test coverage, some with 100% coverage, and many that fall somewhere in between. We should require all source files to have at least 90% coverage. An initiative should be kicked off to write more tests so that all existing source files are 90% covered.”

☐ Come up with a plausible sounding rationale for the CFO:
  • “I think the amount of time (& therefore money) spent fixing bugs that would have been caught is significant. I think the cost of writing the tests now will be less than the money we will save in the future if those tests are added.”

The CFO is now listening to you
How to win over your CFO in 6 easy steps!

Step 2: Collect data and see if a statistically significant relationship exists

- Extract raw data from:
  - source code
  - testing framework
  - version control system
  - issue tracking system, etc.

- Set up regressions:
  - Dependent variable = File-level defect count per unit time
  - Predictor variable = Test-coverage % for each file
  - A variety of control variables

- Demonstrate statistically significant correlation

The CFO is still listening to you
How to win over your CFO in 6 easy steps!

Step 3: Get predicted defect counts for the system as it currently is

- Use your calibrated model generate ‘predicted defect counts’
  - Use model simulation or prediction algorithms
  - Use actual test coverage values and controls as inputs
  - Capture ‘predicted defect counts’ for each file.
  - Sum all of these ‘predicted defect counts’ (per file) to get a ‘predicted defect count’ for the codebase as a whole.

The CFO does not care about this. Do not bother.
How to win over your CFO in 6 easy steps!

Step 4: Get predicted defect counts for your hypothetically improved system

- Modify the input data to set all ‘test coverage values’ to 90% or above
- Leave all other values the same
- Rerun calibrated model with new inputs to generate ‘hypothetical defect counts’

\[ \text{Defect Reduction} = 'predicted defect count' \text{ (from step 3)} - 'hypothetical defect count' \text{ (from step 4)} \]

If ‘Defect Reduction’ is big, the CFO is listening to you again
How to win over your CFO in 6 easy steps!

Steps 5: Estimate annual savings
- Construct reasonable estimates for the cost of increasing test coverage.

- Simple example:
  - It will take a developer 1 day to add coverage to 100 lines
  - 200,000 uncovered lines
  - 200 days in a work year
  - 10 FTE-year to complete
  - $100,000/yr salary
  - $1,000,000 dollars to achieve

Steps 4: Estimate cost of improvement
- Construct reasonable estimates for the annual savings

- Simple Example:
  - Hypothetical model (step 3) predicts 1000 fewer bugs/year
  - Average bug takes 4 days to fix
  - 4000 days saved annually
  - 20 FTE saved annually
  - Breakeven point in 6 months!

The CFO might actually be starting to like you now
How to win over your CFO in 6 easy steps!

Step 6: Estimate the ROI of hypothetical improvement

- **Some Equations:**
  - ‘Cost to fix’ (now) = $1,000,000
  - ‘Annual savings’ = $1,000,000
  - **PRESENT VALUE** of ‘annual savings’ = $3,790,258
    - ((assuming 5 year time horizon and opportunity cost of 10%)
  - ROI = [PRESENT VALUE(annual savings) – ‘cost to fix’] / ‘cost to fix’
  - ROI = 279%

The CFO might even respect you now.
What if you could get quality & refactoring funded?

<table>
<thead>
<tr>
<th>Engineer Suggests</th>
<th>Market Demand</th>
<th>ROI</th>
<th>CFO Reaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Features</td>
<td>• Existing customer will buy more</td>
<td>$</td>
<td>Ecstatic</td>
</tr>
<tr>
<td></td>
<td>• New customers created</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Product differentiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bug Fixes</td>
<td>• Existing customer screaming</td>
<td>$</td>
<td>Required, unfortunately</td>
</tr>
<tr>
<td></td>
<td>• New customers avoiding</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Reputation hit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internal Quality Improvement,</td>
<td>• Internal modeling used to compare ROI and</td>
<td>$$</td>
<td>Tell me more</td>
</tr>
<tr>
<td>Refactoring Initiatives</td>
<td>breakeven point against other investment</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>opportunities</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Introducing Amadeus
2. Challenges faced by the software enterprise
3. Introducing Silverthread, Inc.
4. Case 1: Driving software platform design quality
5. Case 2: Analytics for managerial decision-making
6. Lessons for your company
## Feedback

<table>
<thead>
<tr>
<th>DEVELOPERS</th>
<th>TECHNOLOGY LEADERS</th>
<th>EXECUTIVES</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Sparked architecture discussions</td>
<td>• Helped evaluate the reality against the design intents</td>
<td>• Will provide an objective picture of the portfolio</td>
</tr>
<tr>
<td>• Helped reveal anti-patterns</td>
<td>• Spotted risky areas</td>
<td>• Will allow benchmarking</td>
</tr>
<tr>
<td>• Supported their overall perception with data</td>
<td>• Built business case for refactoring actions</td>
<td>• Will enable arbitration and steering based on ROI</td>
</tr>
</tbody>
</table>

Everyone speaks the same language now
Using the concept for continuous improvement

1. Building insight
   Helping developers do their job
   - Understand system better:
     - Architecture maps
     - System Metrics
     - Project Metrics
     - Software economics

2. Learning
   Connecting technical & economic choices
   - Identify and explore challenges:
     - Code quality
     - Test quality
     - Design quality
     - Software economics

3. Decision making
   Objective business decisions
   - Explore software economics via statistics and modeling
   - Model the ROI of development initiatives
   - Make investments in quality improvement

4. Effective steering & management
   - Measure and enforce architecture rules
   - Set KPIs for improvement initiatives
   - Prioritize and manage improvement
   - Learn and revise models as appropriate
   - Propose new things to explore
Questions ??

Martin Jouvenot  
Software Architect  
Amadeus  
(& MIT SDM ’17)  
mjouvenot@amadeus.com

Rashesh Jethi  
Head of R&D - Americas  
Amadeus  
rashesh.jethi@amadeus.com

Dan Sturtevant  
CEO  
Silverthread, Inc.  
(& MIT SDM Alum)  
dan@silverthreadinc.com