Are You Ready for Agile?

Ian Lawthers

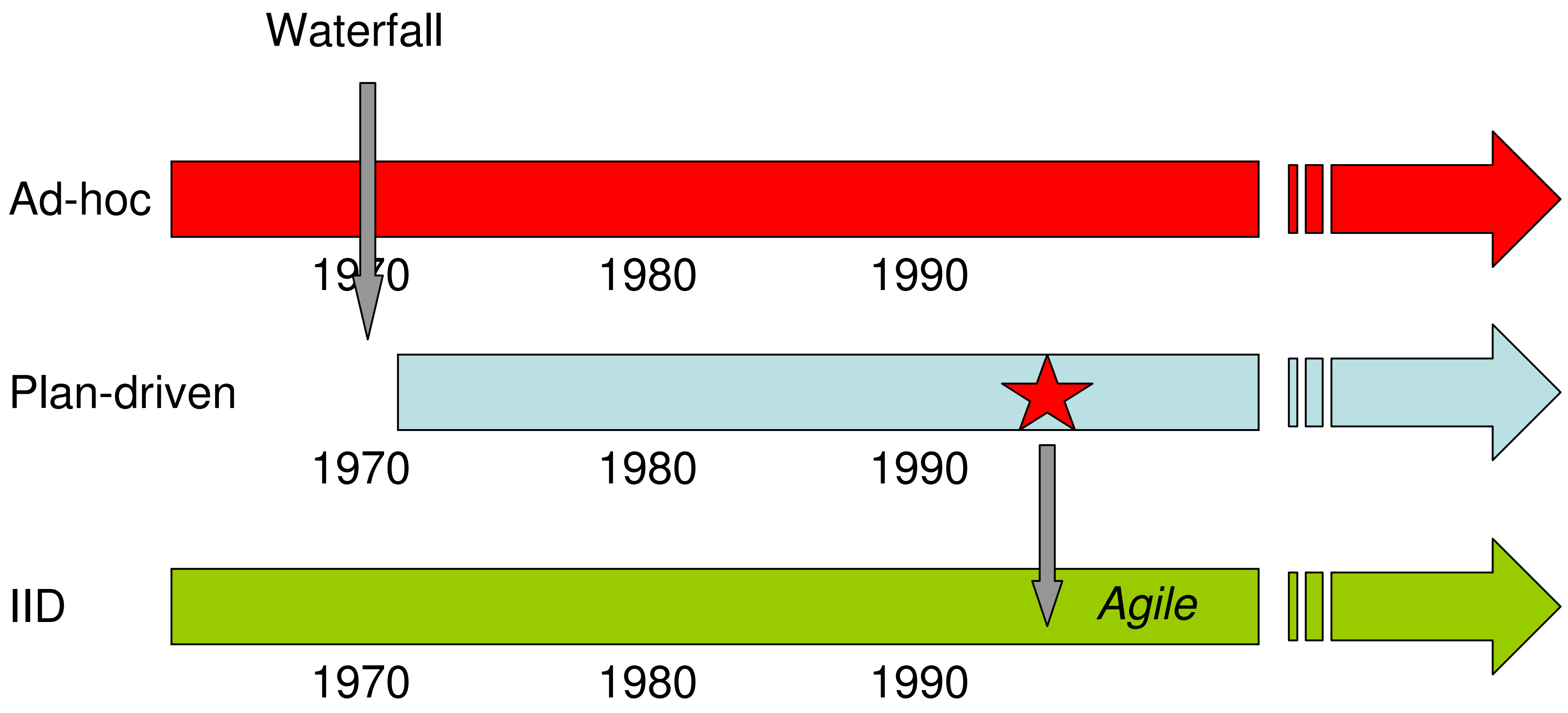
Centre for Software Engineering

Agile – Is it new?

- Agile Methods are a subset of incremental and iterative development (IID)
- IID has been around since the 60's
  - early 60's - NASA Project Mercury ($^1/_2$ Day iterations, Test First)
  - 1972: USA Trident Submarine system (1m LOC, Life Critical, 4 x 6-month iterations)
  - 1977-80: NASA Primary Avionics Software System (Life-critical,17 iterations over 31 months)
  - 1990's: Magnavox Electronic Systems US Army field artillery command and control system (> 1 million Ada LOC, 5 time-boxed iterations)

Not New but re-invented?

- Agile Methods bring a freshness to IID



Agile Manifesto

Individuals and Interactions over Processes and Tools

Working Software over Comprehensive Documentation

Customer Collaboration over Contract Negotiation

Responding to Change over Following a Plan

So what is Agility ?

"Agility is the ability to both create and respond to change in order to profit in a turbulent business environment"

"Agility is the ability to balance flexibility and stability" (Highsmith 2002)

Characteristics of Agile Methods

- Iterative Development
- Requirements Not Fully Understood
- Requirements Change is the Norm
- New Tools / Technologies Make Process Unpredictable

Agile Methods

- About 10 "Agile Methods" since mid 90's
  - Extreme Programming (XP)
  - Scrum
  - Feature Driven Development
  - Crystal
  - DSDM
  - (Rational) Unified Process
  - Lean Software Development

- Scrum and Extreme Programming are the best known ones
  - Scrum emphasises project management
  - XP emphasises developer activity
  - Work well together = XP@Scrum)

Extreme Programming (XP)

- Taking things to extreme e.g. if inspection is good, do it all the time = pair programming
- Sample Practices:
  - 1-3 week iterations
  - User Stories for collecting requirements
  - On-site customer
  - Test Driven Development
  - Do the simplest thing possible
  - Refactoring
  - Coding Standards
  - Continuous integration

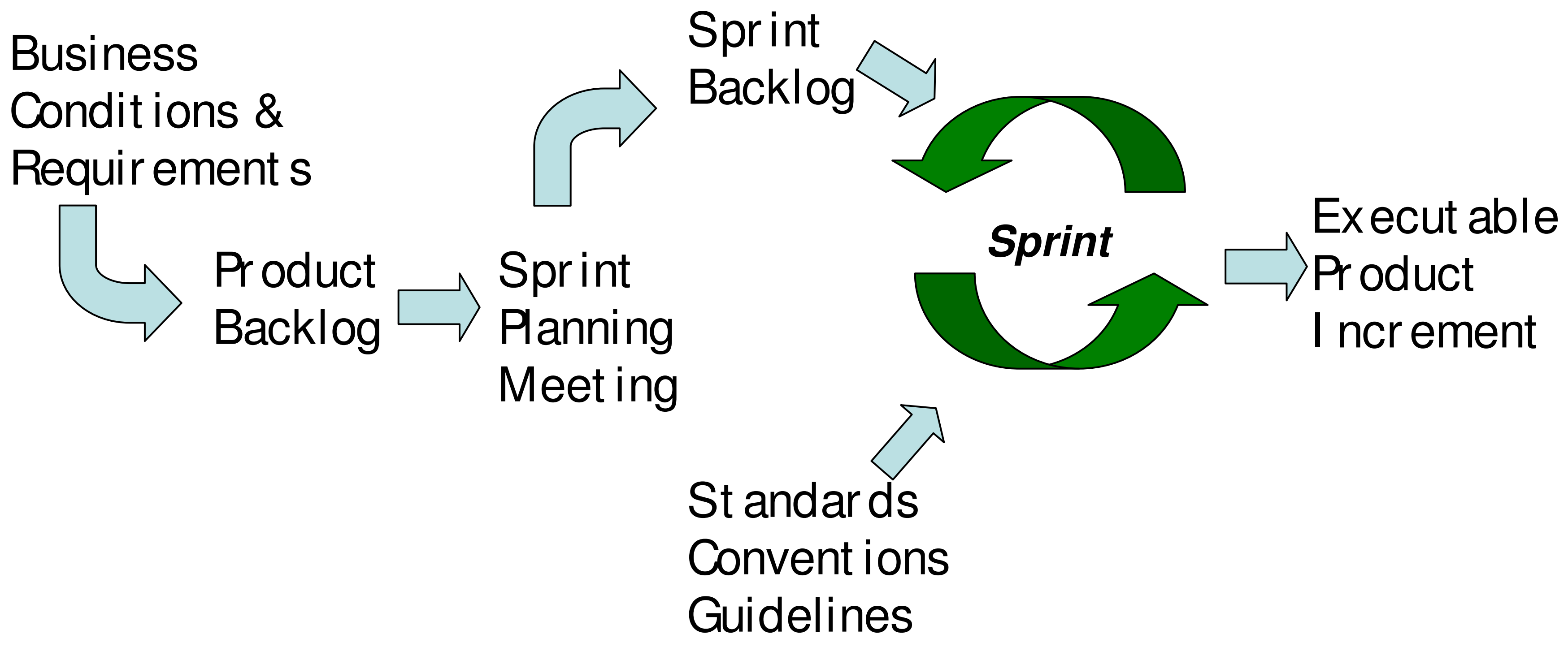
Extreme Programming – Main contributions

- Small Releases
  - Small but big enough to give value
- Pair Programming
  - Driver – writes the code
  - Partner – Thinks about missing tests, integration issues etc
  - Pairs change frequently
- Refactoring
  - Simplifying/ improving code
  - Automated tests check behaviour not changed

Extreme Programming – Main Contributions

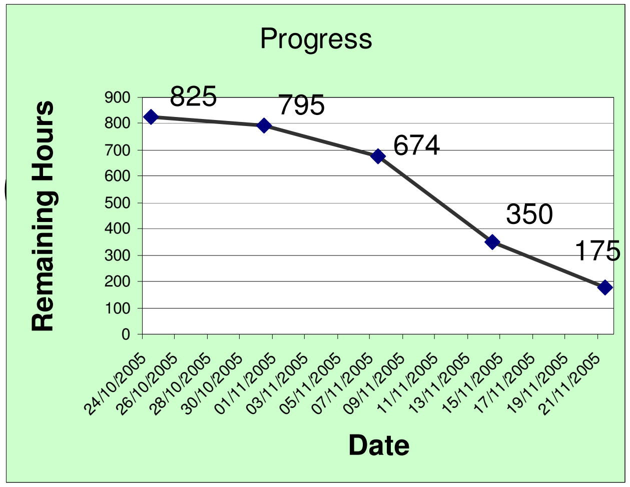
- Test-Driven Development (or Test First)
  - Write unit tests first, before the code
  - Use of Unit Testing Tools
  - Continuous Integration
  - Integration builds at end of day (or even continuously)

Agile Methods – Scrum (1)


Business Conditions & Requirements
Product Backlog
Sprint Planning Meeting
Sprint Backlog
Sprint
Standards Conventions Guidelines
Executable Product Increment

Agile Methods – S

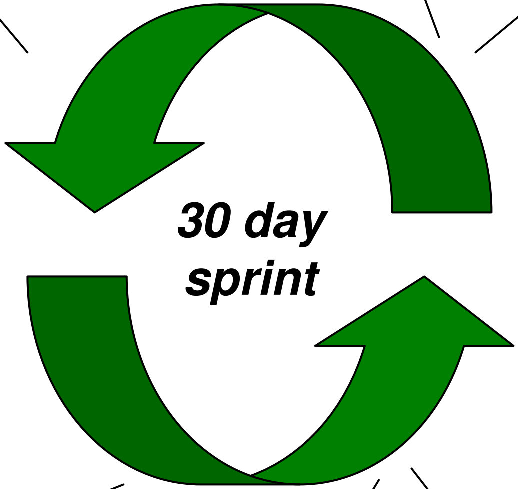
- Scrum development
  int =



Developers sign up for tasks

Sprint backlog graph

No new tasks for 30 days (unless critical)



Daily Scrum meeting

- Same time/place everyday
- Approx 15 mins
- Facilitated by Scrum master
- All team members attend
- Managers do not participate
- Used to raise issues and obstacles only

What have you done since last scrum?

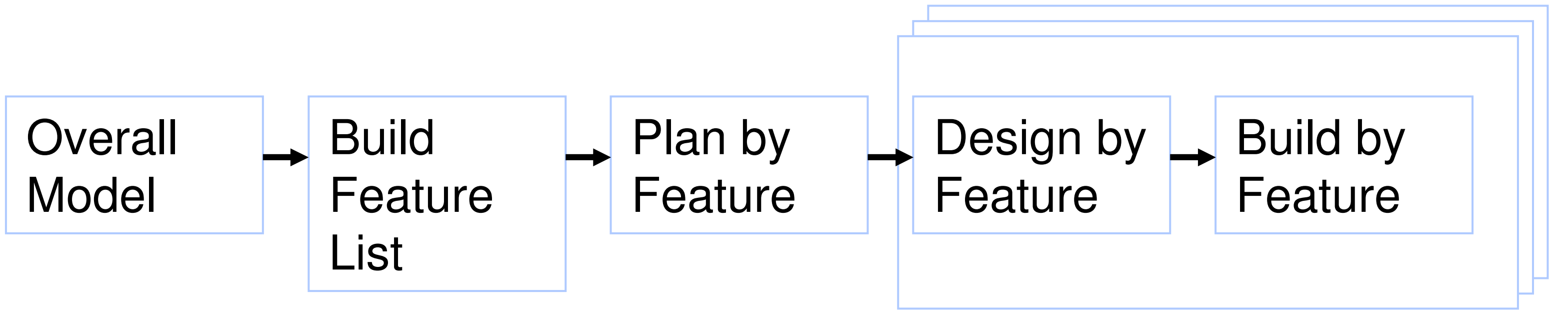
What will you do before next scrum?

What is blocking you?

Any new tasks for Sprint?

Other Agile approaches

- Feature Driven Development (Coad)
  - Unit of work is features, grouped into feature set



- DSDM – based on RAD, has been called ‘Agile’
- RUP – can be used in an agile approach
- AUP, Open UP, Skinny UP

Moving to the Agile Approach Case Study

Agile & Case Study

It has traditionally used a Waterfall Model

- it was beginning to creak
- traditional delivery cycles of 9 months or longer.
- A couple of high profile failures

Agile formally introduced 18 months ago

- 90-day release cycles
  - starting with intensive workshop
  - address specific business opportunities or problems.
- It has become mandatory
  - different interpretations of Agile
  - still some scepticism

Core Agile Practices

1

Iterative Development - Agile projects base delivery of software and other project outcomes around fixed periods of time called Iterations within the 90 day cycle.

Replaces:

- Phased Development
  (ie requirements then design then code then test)

Benefits:

- Early testing and deployment
- Easy adaptation to changing priorities

Core Agile Practices

2

Automated Testing - The underlying principle is the efficient delivery of timely feedback by doing testing as early as possible and as quickly as possible.

Replaces:

- Manual Testing

Benefits:

- Efficient Delivery of Timely feedback
- Automated Unit & Acceptance Tests providing greater than 80% code coverage

Core Agile Practices

3

Continuous Integration - fully automated build and test process that allows a team to build and test their software many times a day.

Replaces:
- Testing and Integration at the end of Delivery Process

Benefits:
- Allows changes in requirements or structural improvements to be safely incorporated into the software
- If the entire system is re-validated after every small change, then it is easy to identify the cause of any issue and resolve it

Core Agile Practices

4

User Stories – The User Story is the basic unit of scope in an Agile project and describes the who, what, why of a requirement

Replaces:

- Long and elaborate requirements documents

Benefits:

- Very effective mechanism for decomposing requirements into prioritised, testable, estimatable bite-sized pieces of work that the customer can touch & feel

Core Agile Practices

5

Customer Involvement - Agile methods consistently emphasise ongoing involvement of the Customer with the IT team throughout the cycle, providing constant input and feedback.

Replaces:

- Customer expectations not managed through Development process

Benefits:

- Ongoing involvement of the Customer with the delivery/development team throughout the iterations, providing input and feedback ensuring the customer gets what is needed

Other Recommended Agile Practices

- **Retrospectives**
  - at least one per project per cycle (3 to 5 hours)
  - an informal one after each iteration (< 1 hour)
- **Pair programming**
  - not just for less experienced developers
  - for trouble shooting
- **Test Driven Development**
  - test / code / refactor
- **Colocation where possible**

What does Agile mean for Project Managers ?

Project Managers become Scrum Masters

- Gantt charts to Burndown charts
- PM estimates to Team estimates
  - » *"a Volunteer is worth two Conscripts"*

Scrum Values

- Commitment - defined goal per iteration
- Focus - no distractions
- Openness - daily scrum meetings
- Courage - confidence to take responsibility

The Scrum Master sets the Team Ethos and is the guardian of the values

What does Agile mean for Project Managers ?

PMs run the Planning Game

- Release Planning - 90 day delivery with Retrospective
- Iteration Planning - 2 weeks with short review
- User Stories - estimated in days
- Tasks - estimated in hours

PMs gather the Estimates

- Stories & Tasks estimated in perfect days
- Experience dictates how many perfect days per iteration
- Team buy-in

They track using Burndown Charts

- Tracking estimated work still to do
- Tracking time spent on stories / tasks

They remove obstacles

What does Agile Mean for Software Engineers ?

- **Focus on the delivering deployable code**
- **Focus on automation**
  - deployment
  - testing
- **Greater Use of Tools**

Learning points in Implementing Agile

You need people who:

- Are good at Teamwork and collaboration
- Think Simple is good
- Are Self motivated
- Can cope with uncertainty
- Accept responsibility
- Are Adaptable
- Are Technical skilful
- Will focus on making the customer happy

Agile is mainly about people & relationships

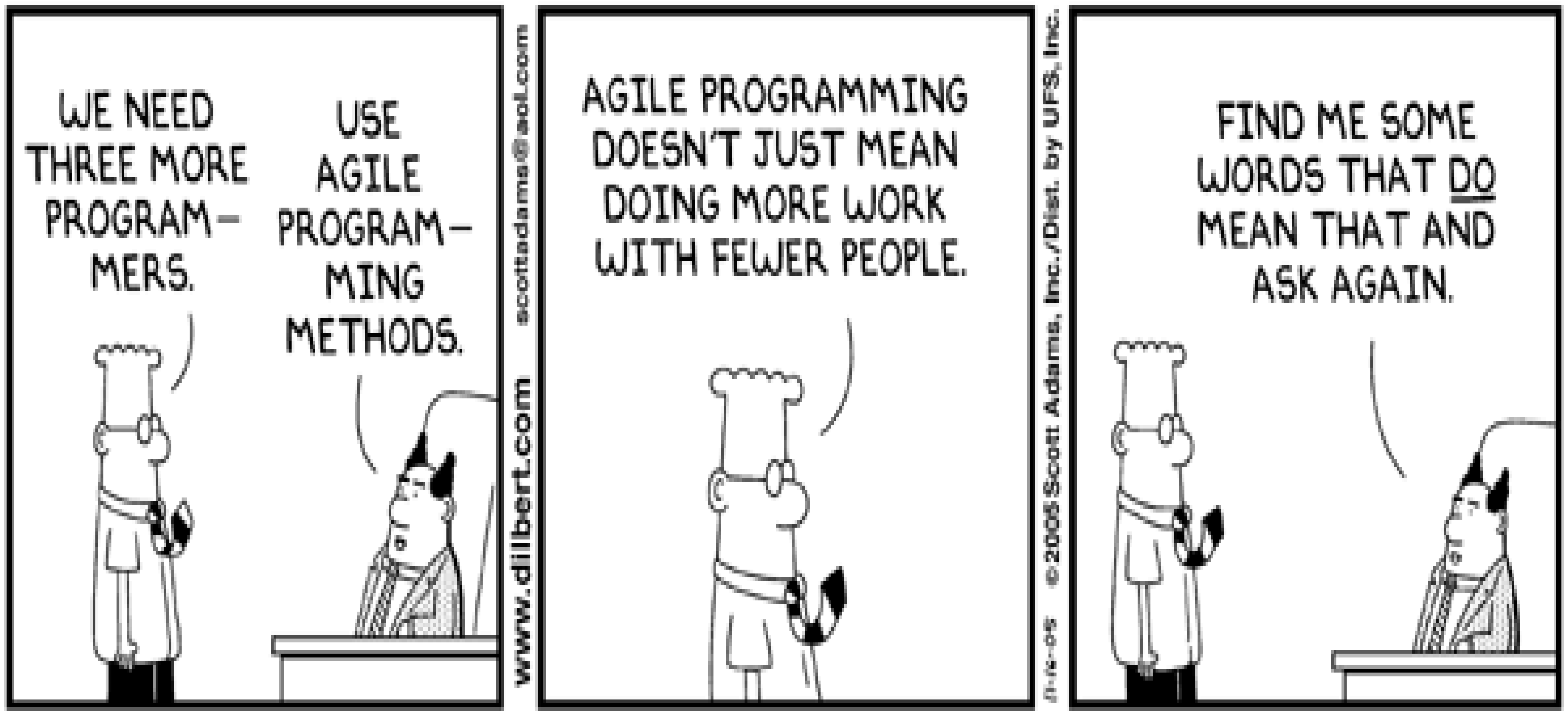
Learning points in Implementing Agile

- **Delivery has improved**
  - cheaper / faster / better tested code
  - risk areas addressed early
- **The teams look and behave differently**
  - fewer managers
  - happier developers
  - collective ownership
- **We deliver working code to our customers**
- **Increased developer discipline**
- **System Test & Design Assurance subsumed into the team**

Learning points in Implementing Agile

- **Training is a must**
  - Developer training & Management Training
  - Mentoring
  - use of consultants
- **Interfacing with non Agile teams/processes**
  - traditional Programme Management
  - end to end design
  - non Agile component teams   e.g. Offshore
  - non Agile deployment

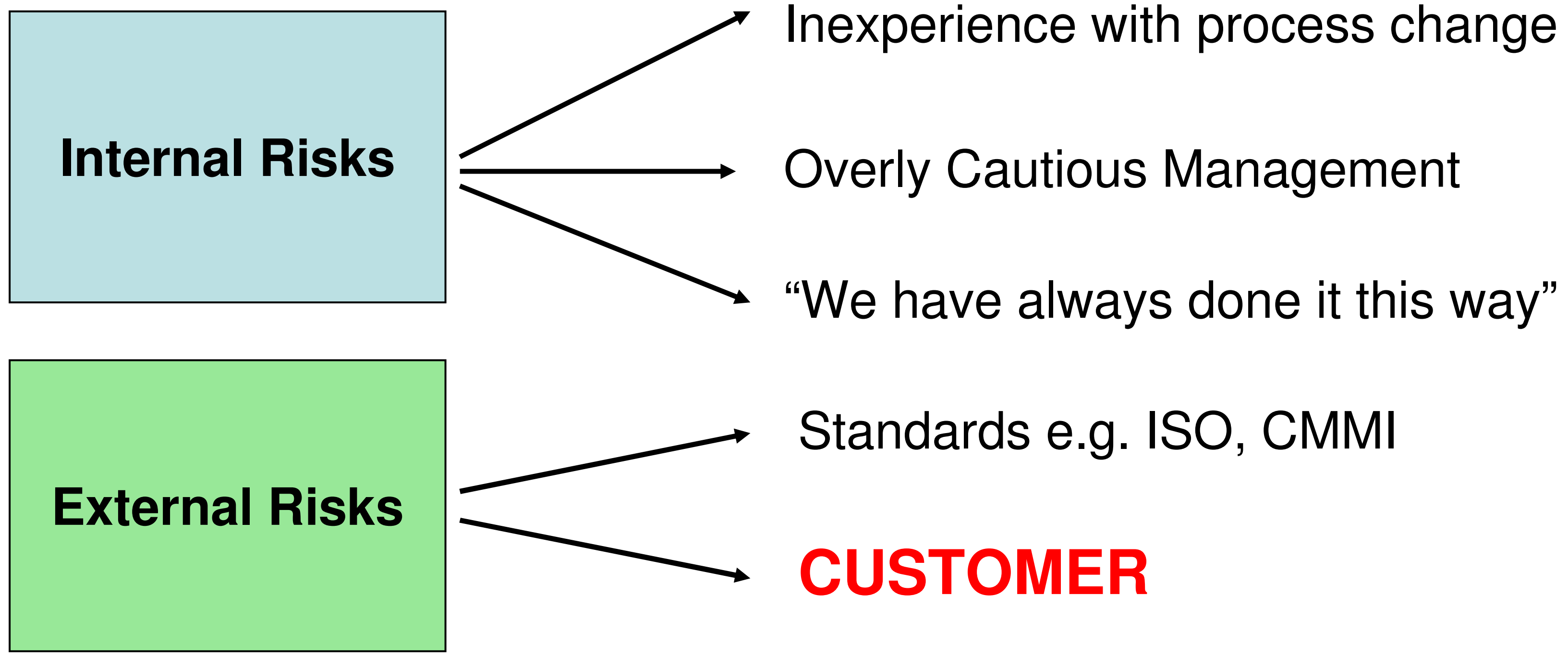
Everyone needs to understand their responsibilities


WE NEED THREE MORE PROGRAM- MERS.
USE AGILE PROGRAM- MING METHODS.
AGILE PROGRAMMING DOESN'T JUST MEAN DOING MORE WORK WITH FEWER PEOPLE.
FIND ME SOME WORDS THAT DO MEAN THAT AND ASK AGAIN.
scottadams@aol.com
www.dilbert.com
© 2005 Scott Adams, Inc./Dist. by UFS, Inc.

Agility/Discipline Assessment

Adopting Agile Methods

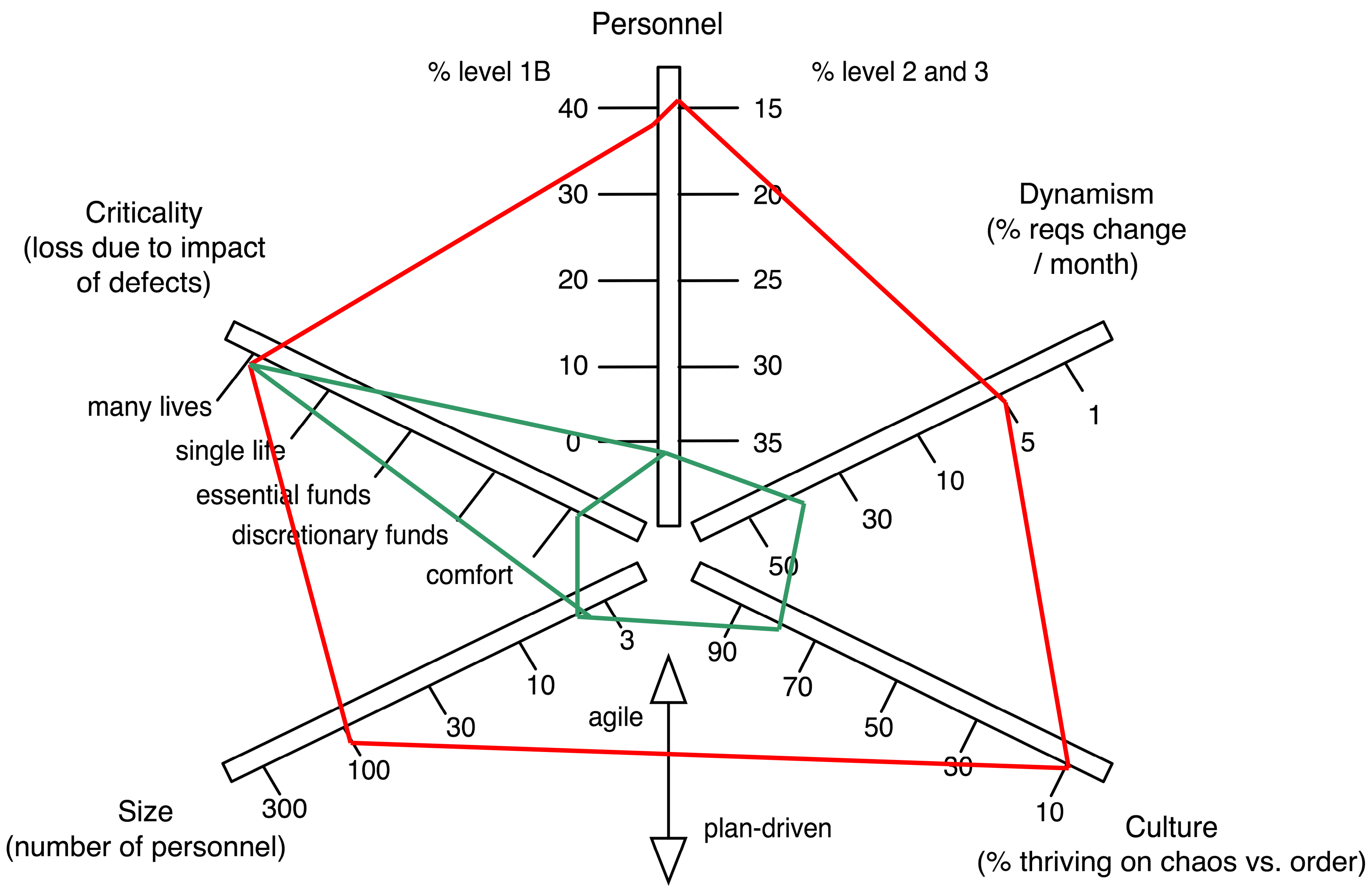
- Any process change can be risky


Internal Risks
Inexperience with process change
Overly Cautious Management
“We have always done it this way”
External Risks
Standards e.g. ISO, CMMI
CUSTOMER

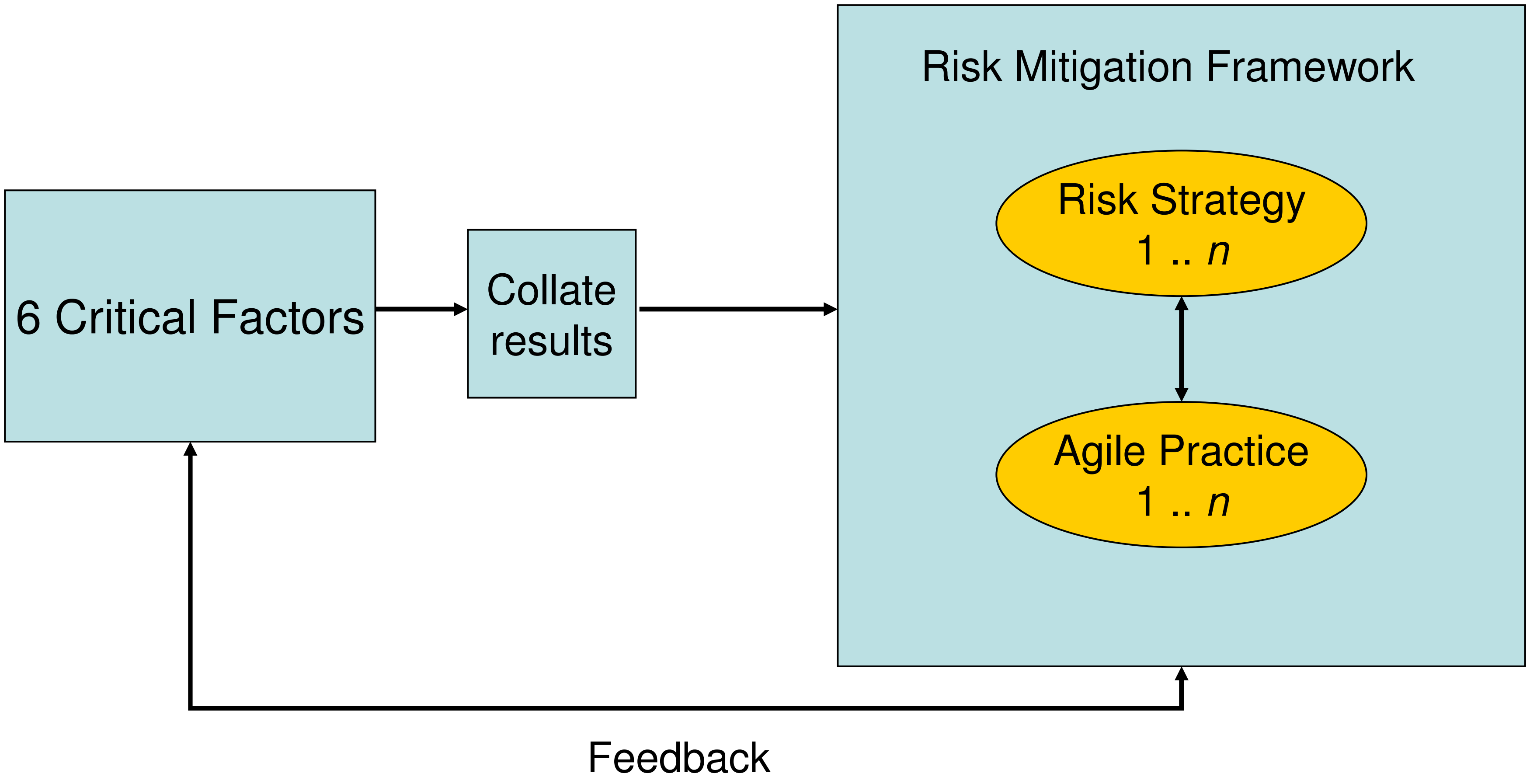
Agility Assessment

- Aimed at Determining how Agile or Defined your Software Process needs to be
- Assessment is Done in Two Stages
- Assessment Filled in by Each Project Team Member & is Confidential
- Does not Provide a Definitive Statement on Suitability for Agile Approach!!

Adopting Agile Methods – Critical Factors



Summary of Approach



Agility/Discipline Assessment
Stage 1 - Critical Factors

Stage 1 Critical Factors

- Looks at Factors that need to be Mitigated for Agile vs Plan-driven Approach
- Consists of Six “Critical Factors”
- Each Project Team Member Plots Own Risk Graph

Personnel

- Each Project Team Member Rates Themselves & the Team
- Level 3: Able to Revise a Method to Fit an Unprecedented Situation
- Level 2: Able to Manage a Precedented Project but Needs Help with Large/Unprecedented Project

Personnel

- Level 1A: Can Perform Agile Development Tasks when Trained
- Level 1B: Can Perform Procedural Tasks when Trained
- Level -1: Technical Skills but Unwilling to Collaborate and/or Follow Shared Methods

Requirements Churn

- Shown as % Requirements Change / Month
- Use Project Metrics (if available)
- Otherwise Estimate

Culture & Team Size

- Organisation Culture
  - Estimate % Thriving on Uncertainty
  - Vs Predictability
- Team Size
  - Number of Personnel on Project Team

Criticality

- Examines Safety Criticality of System
  - Comfort : Minor Problems
  - Discretionary Funds : Cause Business Problems but Can Work Around
  - Essential Funds : Cause Major Problems or Bankruptcy to Business
  - Single Life : Could Cause Death / Serious Injury to an Individual
  - Many Lives : as above but Many Lives

Client Involvement

- Role of Customer in Process
  - AB On-site : Agile Believer & On-site with team
  - AB Off-site : Agile Believer, not On-site but Understands Agile Approach
  - AS On-site : Agile Sceptic & On-site with team, not Bought into Agile Approach
  - AS Off-site : as Above but Client Off-site
  - Uninvolved Off-site : Client Not Involved in Providing Initial Requirements to Ensure Correct Product Delivered

Agility/Discipline Assessment Stage 2 – Risk Assessment

Stage 2 Risk Assessment

- Looks at Principal Risks that may Affect a particular Project
- Consists of Three Categories of Risk
  - Environmental
  - Risks of Using Agile Methods
  - Risks of Using Plan Driven Methods
- Each Project Team Member Rates Each Risk on a 1 (Minimal) – 5 (Showstopper) Scale

Environmental Risks

- Risks From Project’s General Environment
- Three Types
  - Technology Uncertainties
  - Many Stakeholders
  - Complexity of System

Risks of Using Agile Methods

- Risks Specific to Agile Methods
- Four Risks
  - Scalability & Criticality
  - Use of Simple Design
  - Personnel Turnover
  - Lack of Skilled People in Agile

Risks of Using Plan Driven Methods

- Risks Specific to Plan-Driven Methods
- Four Risks
  - Rapid Change
  - Need for Rapid Results
  - Emergent Requirements
  - Lack of Skilled People in Plan-Driven Methods

The Risk Dimensions are not…

- An exact measure of your project
- The only task you should do when deciding how much agility or discipline you need

The Risk Dimensions are…

- A useful framework for introducing agile
- Helpful for promoting discussion about the effectiveness of agile

Agility/Discipline Assessment Case Study – Company 1

Company 1

- Small Indigenous Software Company
- Supplier of Software and other Services to Sports Industry
- Software Personnel
  - Four Developers
  - Graphic Designer
- Projects 10-12 Weeks
- No Process in Place

Controlled by Contracts

- Types of contracts

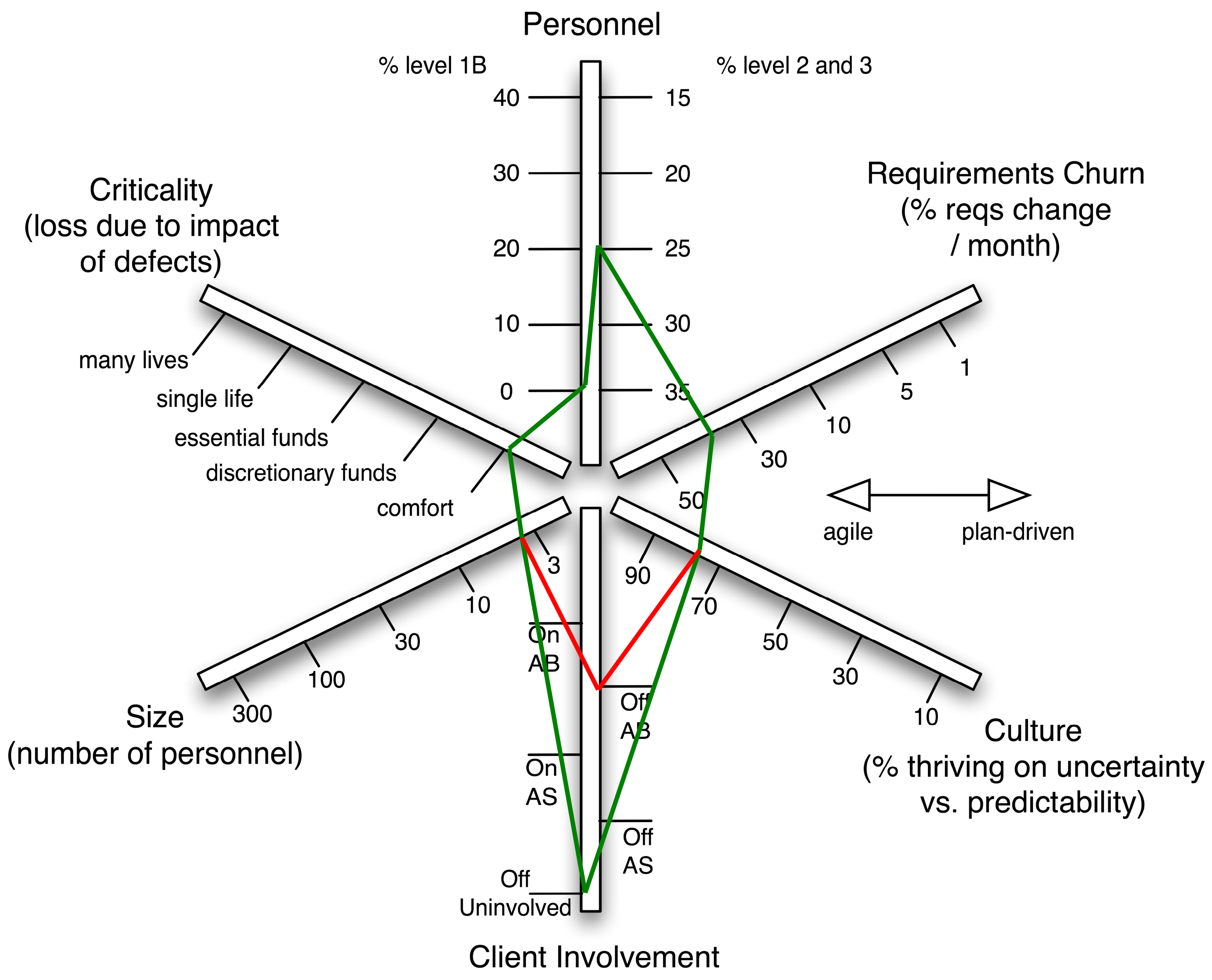
Fixed Price

Favour the customer

Lack of trust?

Vendor protects themselves with detailed spec

Company 1 Agile Criticality



Customer Collaboration

- The aim needs to be:

Customer collaboration *over* contract negotiation

Engaging a Customer (1)

- Try to get them to be Off-Site Agile Believers
- Implemented weekly incremental delivery for the last three weeks
- Final increment = handover release
- Customer involvement in incremental release is contractually required

Engaging a Customer (2)

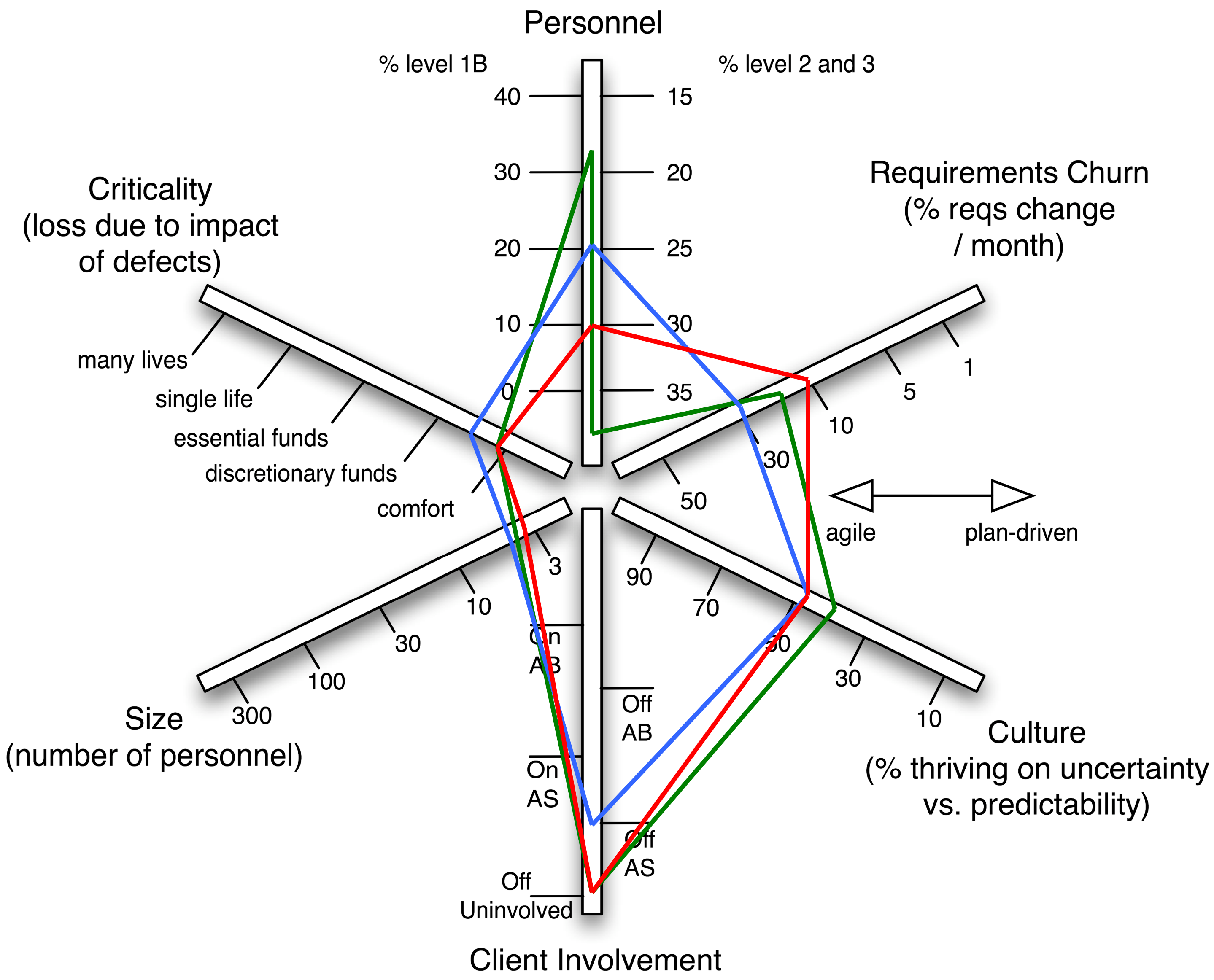
- User acceptance tests fundamental to incremental releases
- User acceptance tests in language of the customer
- Increments result in failed user acceptance tests and/or new requirements
- Customer and Company decide what needs to be completed in next increment or new increment

Agility/Discipline Assessment Case Study – Company 2

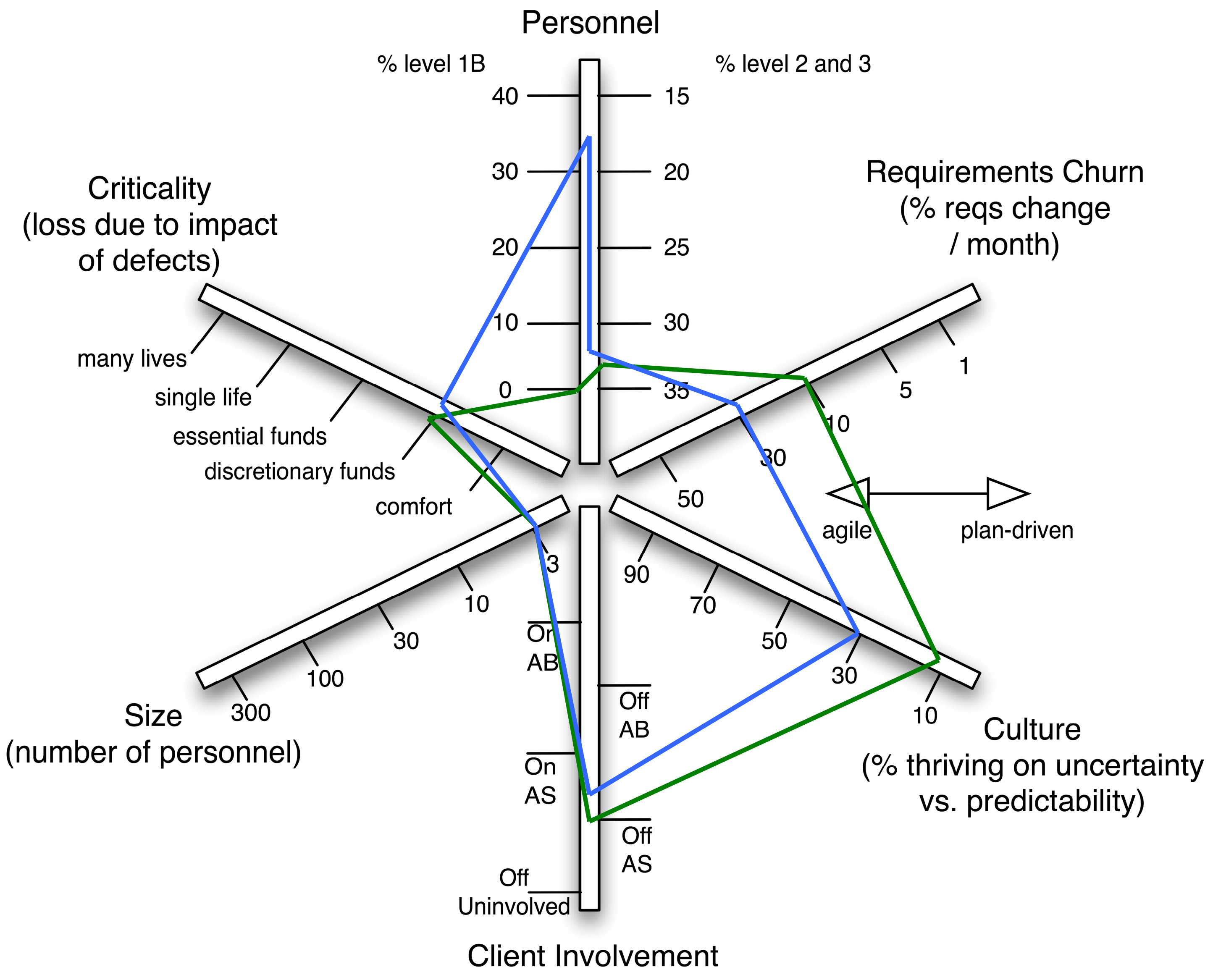
Company 2

- Indigenous Software Company
- Large by Local Standards
- Supplier of CRM Systems
- Main Markets
  - UK, North America
- Main Customers
  - UK Government

Project 1


Personnel
% level 1B
% level 2 and 3
40
30
20
10
0
15
20
25
30
35
Requirements Churn
(% reqs change
/ month)
1
5
10
30
50
agile
plan-driven
Criticality
(loss due to impact
of defects)
many lives
single life
essential funds
discretionary funds
comfort
Size
(number of personnel)
3
10
30
100
300
Culture
(% thriving on uncertainty
vs. predictability)
90
70
50
30
10
On
AB
Off
AB
On
AS
Off
AS
Off
Uninvolved
Client Involvement

Project 2



Personnel
% level 1B
% level 2 and 3
40
30
20
10
0
15
20
25
30
35
Requirements Churn (% reqs change / month)
1
5
10
30
50
agile
plan-driven
Culture (% thriving on uncertainty vs. predictability)
90
70
50
30
10
Client Involvement
On AB
Off AB
On AS
Off AS
Off Uninvolved
Size (number of personnel)
3
10
30
100
300
Criticality (loss due to impact of defects)
many lives
single life
essential funds
discretionary funds
comfort

Personnel Risk Factor

- Suggested Reasons:
  - Personnel Turnover
  - Availability of Team Members
  - Training Required
  - Lack of Documentation
- Possible Mitigation Strategies
  - Personnel Rotation
  - Training
  - Pair Programming / Mentoring
  - Documentation at *Required* Level

Client Involvement Risk

- Clients Often on or off-site AS or Uninvolved
- Mitigation Strategies:
  - Employ Incremental Delivery & Agree With Customer
  - Final Increment is Handover
  - Get Customer More Involved in UAT
  - Write UAT in Language of Customer

Agility/Discipline Assessment
Stage 2 – Risk Assessment Results

Risk Assessment

- Large Variance in all Three Risk Factors
- Varies from Project to Project
- Suggests…
- Some Form of Risk Management Approach would be Useful on all Projects

Risk Mitigation Strategies - 1

- Skilled Practitioners
  - Key Personnel Selection Criteria
  - External Mentor / Contractor
  - Customer Involvement
  - Relevant Training

Risk Mitigation Strategies - 2

- Use of Simple Design
  - Use Within Agile Module Teams
  - Design Patterns
  - Take Design to Level of Detail that Mitigates Risk

CSE & Agile

- Agile Services
  - Development, Assessment, Training & Mentoring
  - http://www.cse.dcu.ie/cse_www/pdf/brochures/agile%20brochure%20v1.0.pdf
- Fundamentals of Agile Project Management
  - http://www.cse.dcu.ie/cse_www/events/agile_fundamentals.html

Any Questions?