

Francois Retief

LARGE SOFTWARE SYSTEMS ON TIME & WITHIN BUDGET

Software is a place where dreams are planted and nightmares harvested, an abstract, mystical swamp where terrible demons compete with magical panaceas, a world of werewolves and silver bullets.

-Brad J. Cox

RECOGNITION AND APPRECIATION



◎ Jaco van der Merwe

Head of Software Development- EMSS

◎ Pierre Hammond

Engagement Manager Microsoft

PRESENTATION OVERVIEW

- My Background
- The challenges of telematics systems development
- Careful, it is not as easy as it seems
- The Power of Agility
- Practical experience in the development of telematics and other complex software systems
- Conclusion

- ◎ My Background
- ◎ The challenges of telematics systems development
- ◎ Careful, it is not as easy as it seems
- ◎ The Power of Agility
- ◎ Practical experience in the development of telematics and other complex software systems
- ◎ Conclusion

MY BACKGROUND

If you hold a cat by the tail you learn things you cannot learn any other way.

-Mark Twain

MY BACKGROUND

- Entrepreneur
- Field & Qualification Engineer
- Systems Engineer
- Production Engineer
- Senior Systems Engineer
- Systems Engineering Manager
- Project Manager/ Systems Engineer
- Consultant/ Contractor

•My Background

- The challenges of telematics systems development
- Careful, it is not as easy as it seems
- The Power of Agility
- Practical experience in the development of telematics and other complex software systems
- Conclusion

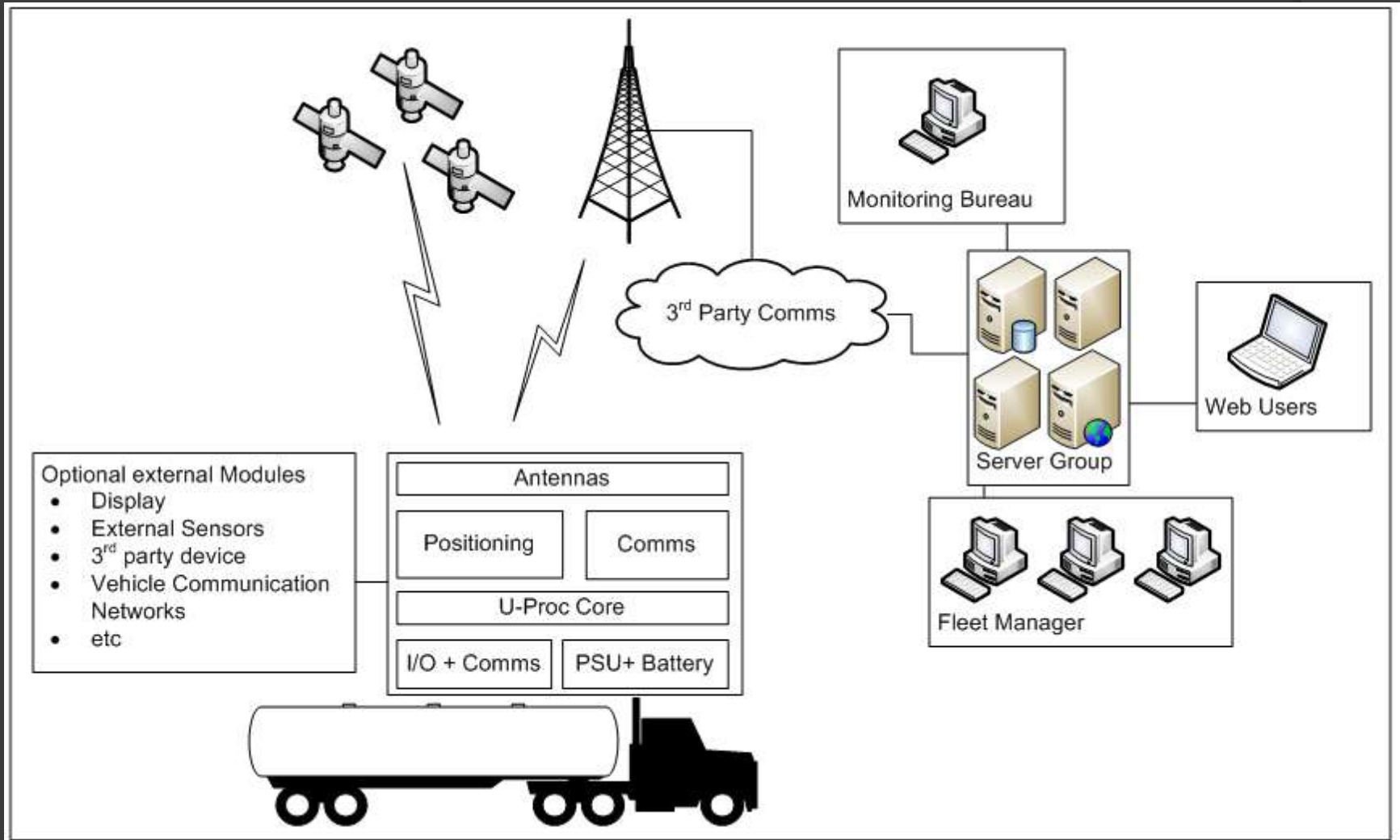
THE CHALLENGES OF TELEMATICS SYSTEMS DEVELOPMENT

- My Background
- The challenges of telematics systems development**
- Careful, it is not as easy as it seems
- The Power of Agility
- Practical experience in the development of telematics and other complex software systems
- Conclusion

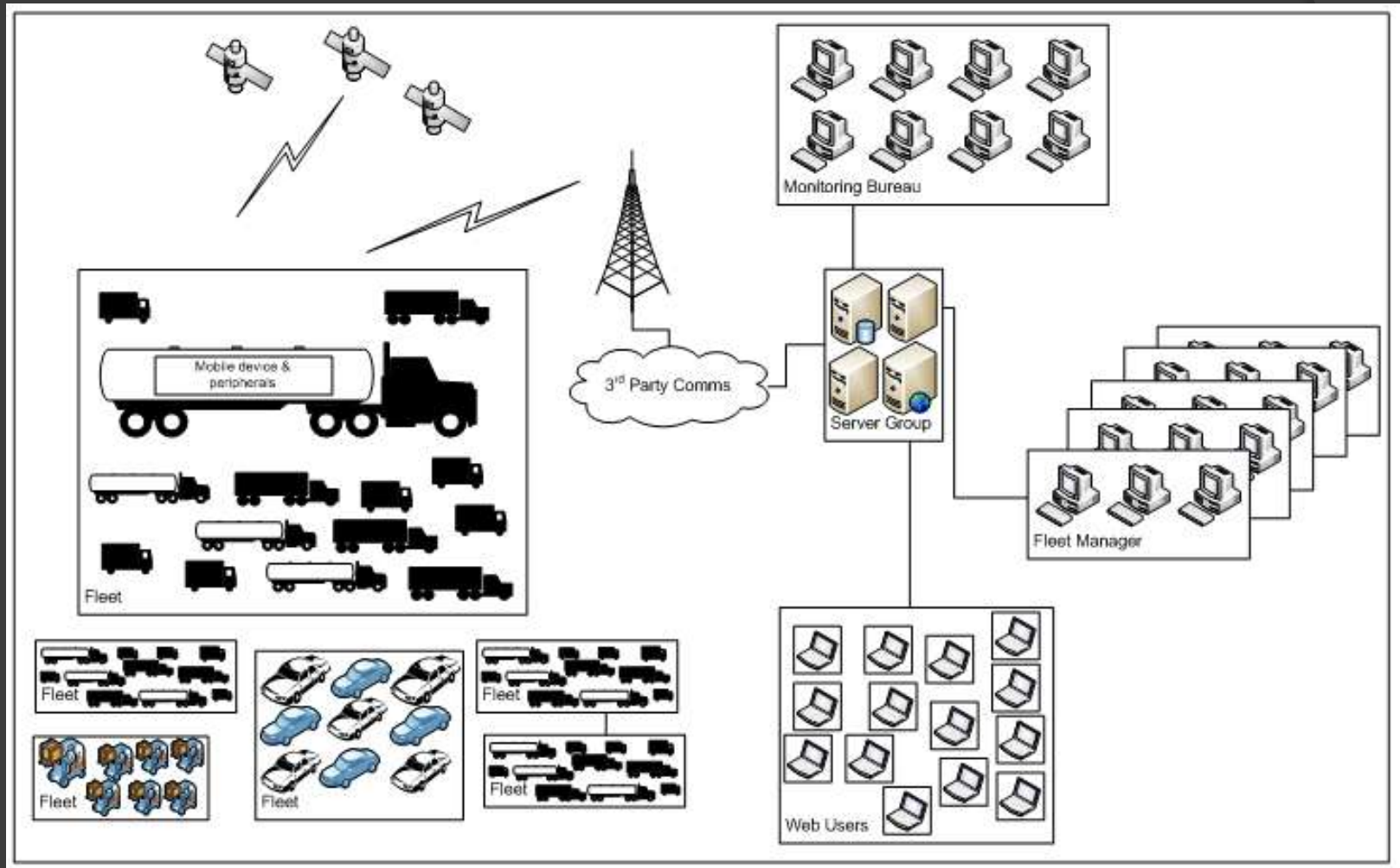
Controlling complexity is the essence of computer programming.

- Brian Wilson Kernighan

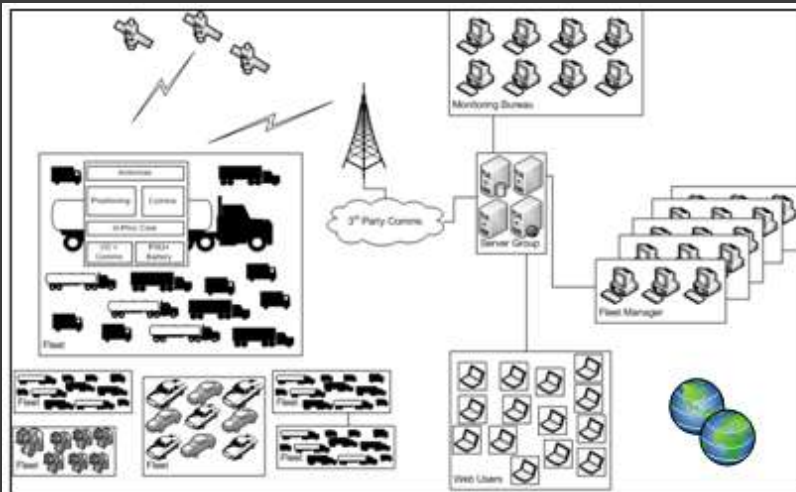
TELEMATICS SYSTEMS



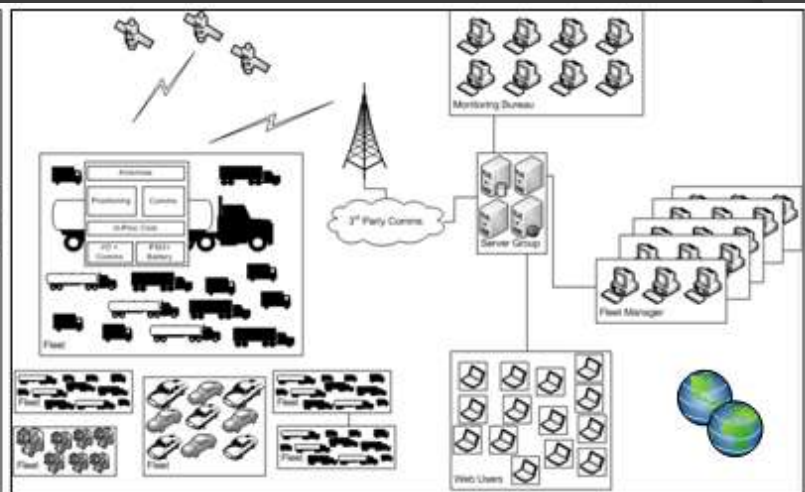
TELEMATICS SYSTEMS (2)



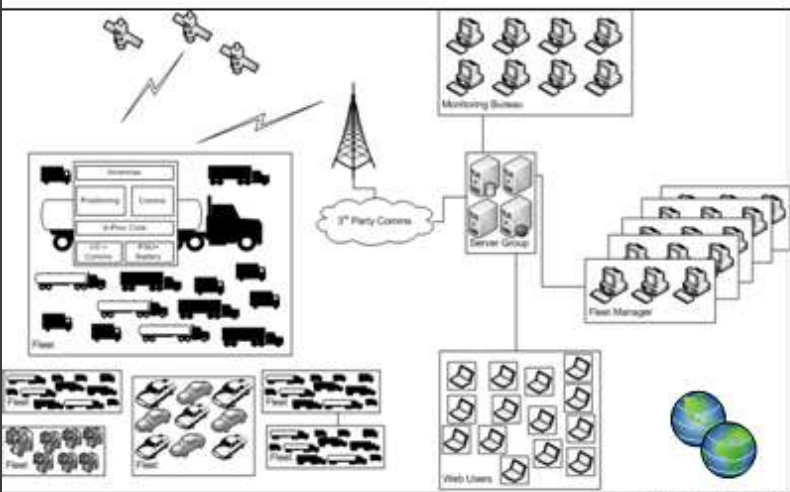
TELEMATICS SYSTEMS (3)



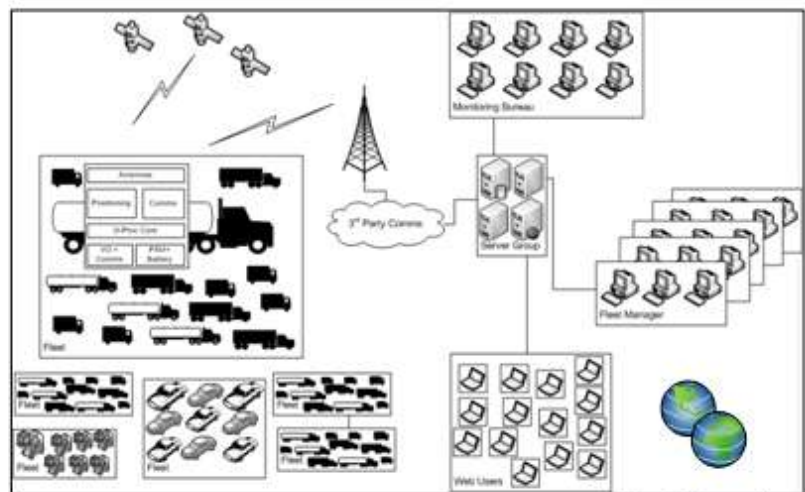
Africa and Middle East



Europe



South East Asia



North America

TELEMATICS SYSTEMS (4)

- 10 years old
- 2-4 releases per year
- 1-2 new hardware products per year
- 1 new platform every 2 to 4 years

CAREFUL, IT IS NOT AS EASY AS IT SEEMS

- My Background
- The challenges of telematics systems development
- Careful, it is not as easy as it seems**
- The Power of Agility
- Practical experience in the development of telematics and other complex software systems
- Conclusion



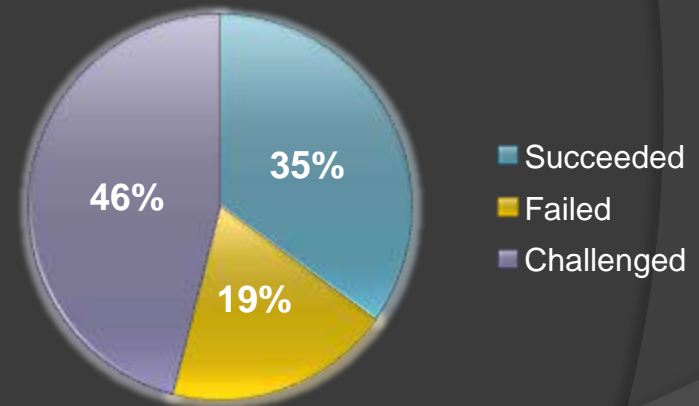
“Good judgement comes from experience, and experience comes from bad judgement.”

-Frederick P. Brooks

CAREFUL, IT IS NOT AS EASY AS IT SEEMS

- ◎ Chaos Survey
- ◎ Standish Group's SURF database
- ◎ 50,000 Software Projects
- ◎ Around the world
- ◎ Since 1994

2006 Chaos Report Summary



CAREFUL, IT IS NOT AS EASY AS IT SEEMS

1994 to 2006 Standish Chaos Report Summary



HOW TO SUCCEED



Top Ten Reasons for Success

- 1. User Involvement
- 2. Executive Management Support
- 3. Clear Business Objectives
- 4. Optimizing Scope
- 5. Agile Process
- 6. Project Manager Expertise
- 7. Financial Management
- 8. Skilled Resources
- 9. Formal Methodology
- 10. Standard Tools and Infrastructure

THE POWER OF AGILITY

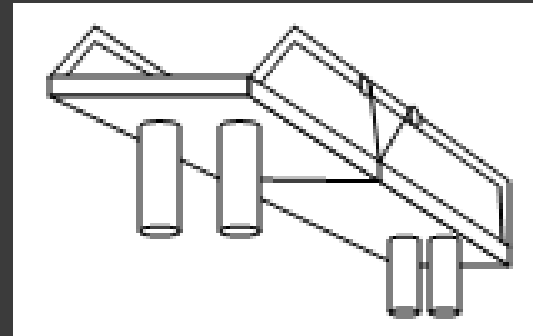
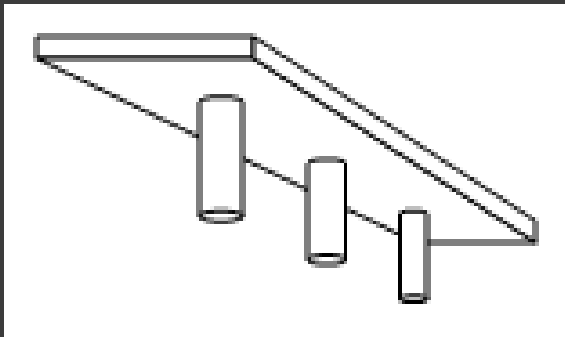
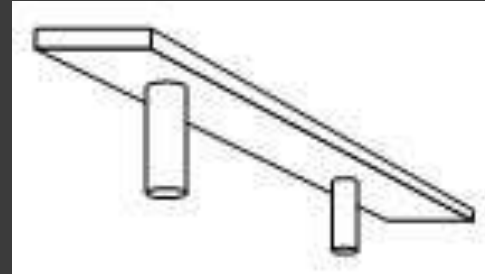
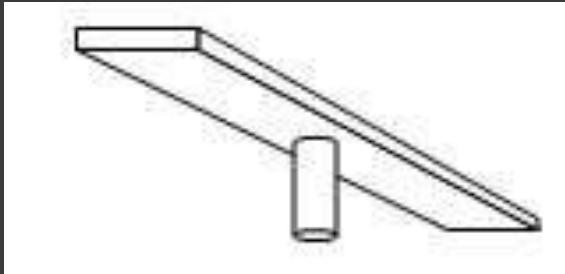
- My Background
- The challenges of telematics systems development
- Careful, it is not as easy as it seems
- The Power of Agility**
- Practical experience in the development of telematics and other complex software systems
- Conclusion



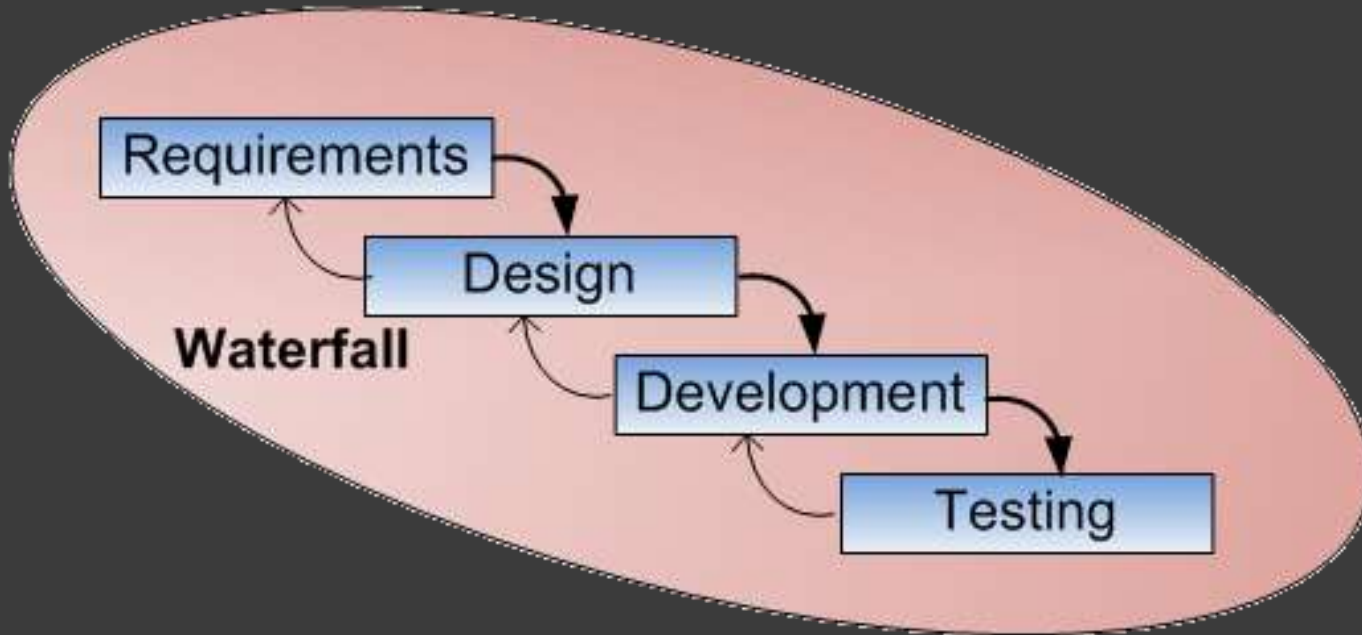
“Successful software always gets changed.”

-Frederick P. Brooks

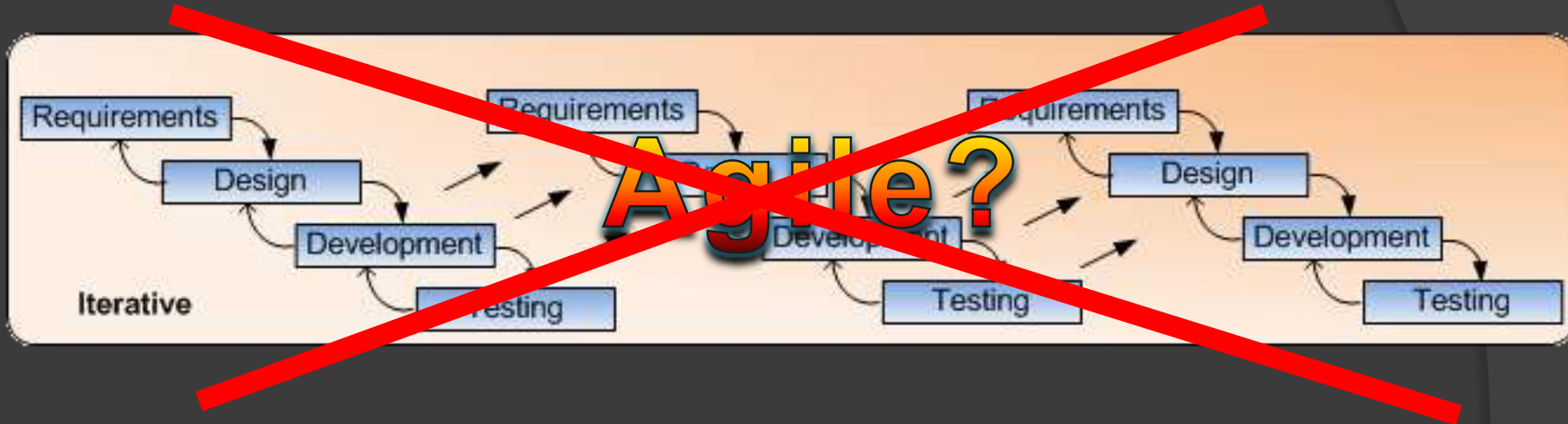
SOFTWARE IS NOT LIKE BUILDING A BRIDGE



WATERFALL METHODOLOGY



ITERATIVE METHODOLOGY



Immediate apparent advantages are:

- Feedback easier and faster
- Integration risk greatly reduced
- Significantly easier to adapt to changing requirements or needs

AGILE DEVELOPMENT

- ⦿ Values + Principles
- ⦿ Management practices
 - Involve Customer Closely
 - Iterate early and regularly
 - Feature estimation and prioritisation
 - Release and iteration planning
 - Velocity
 - Chart Progress
- ⦿ Coding practices
 - Test Driven Design
 - Simple and clean design
 - Refactoring
 - Domain Driven Design

AGILE MANIFESTO

- Individuals and interaction over processes and tools,
- Working software over comprehensive documentation,
- Customer collaboration over negotiation and contracting,
- Responding to change over following a plan.

While the statements on the right are important, the ones on the left are important.

XP

RUP

Evo

FDD

Scrum

Lean Software
Development

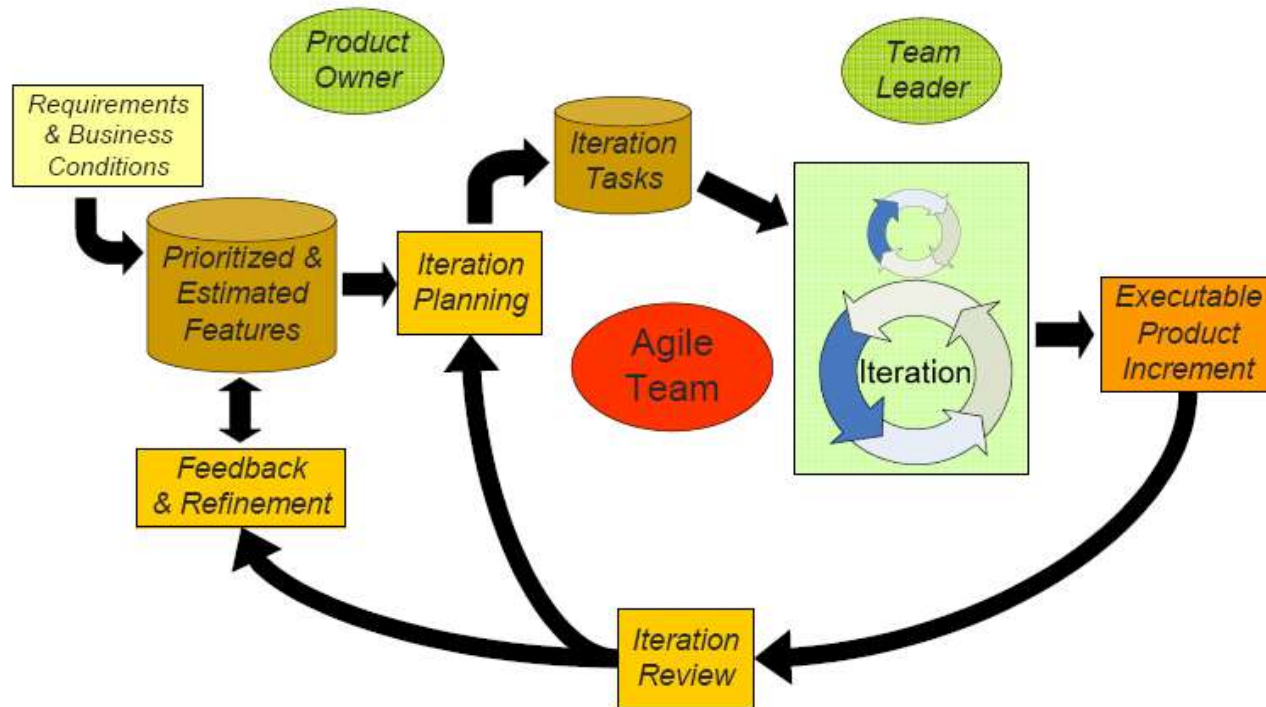
Crystal

KEY DIFFERENCES

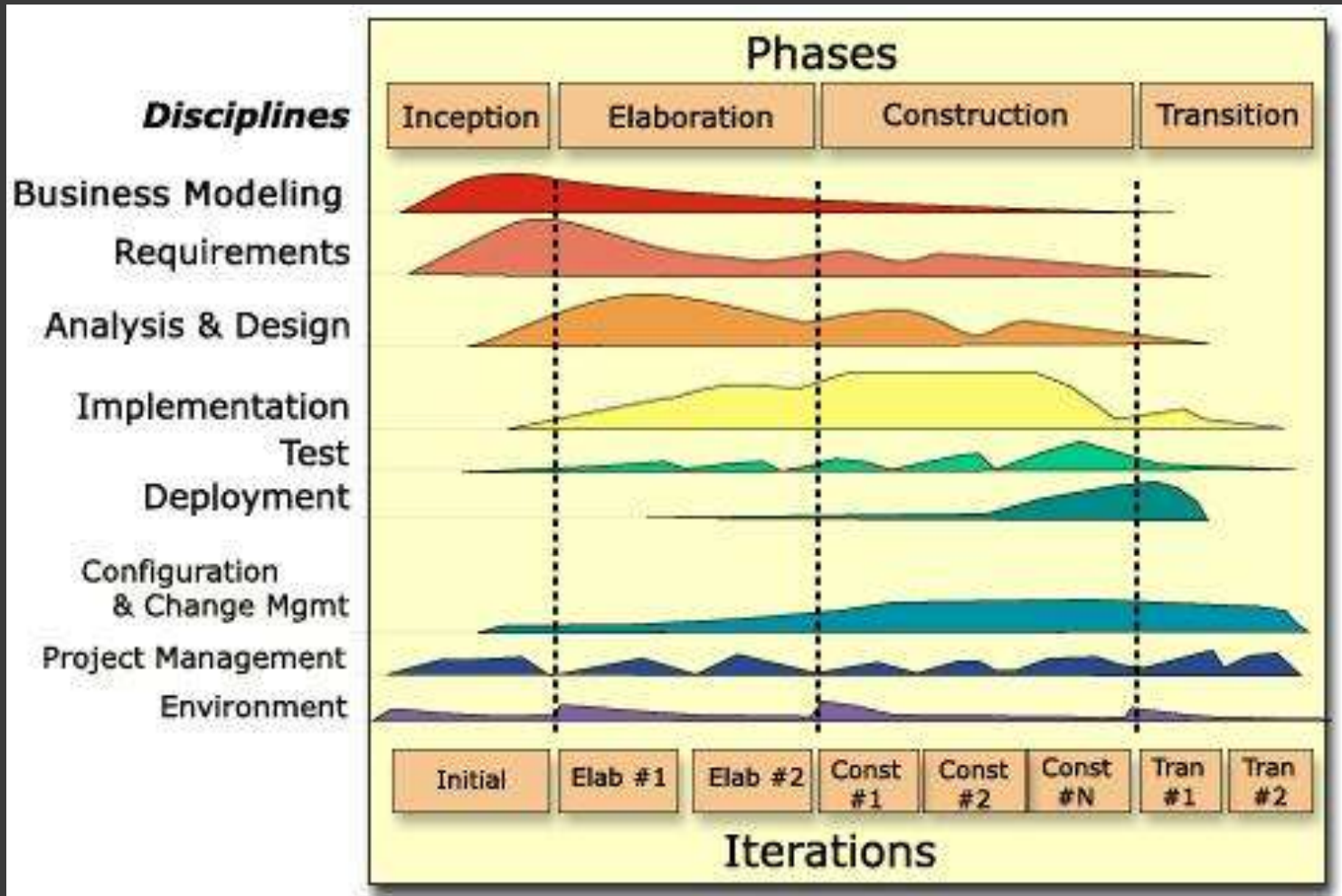
- ⦿ Autonomous, capable, small, self-organising teams with cross functional skills.
- ⦿ Customer part of the development team, solidifying his requirements as the product unfolds.
- ⦿ Upfront Documents less important. Working functions through vertical feature slices. Diagrams and artefacts to document design decisions and capture critical information for future reference.
- ⦿ Notwithstanding a clear vision the team is free to explore and capitalize on emerging possibilities.
- ⦿ Able to focus on risk much earlier (Requirement risk, Integration risk etc.)

PROCESS OVERVIEW

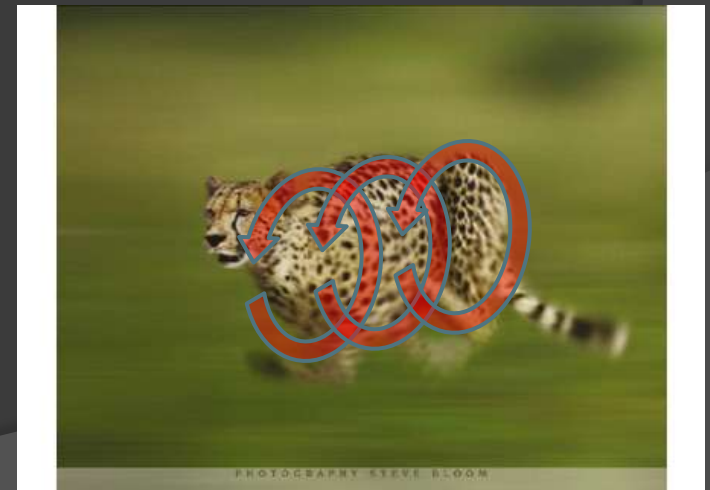
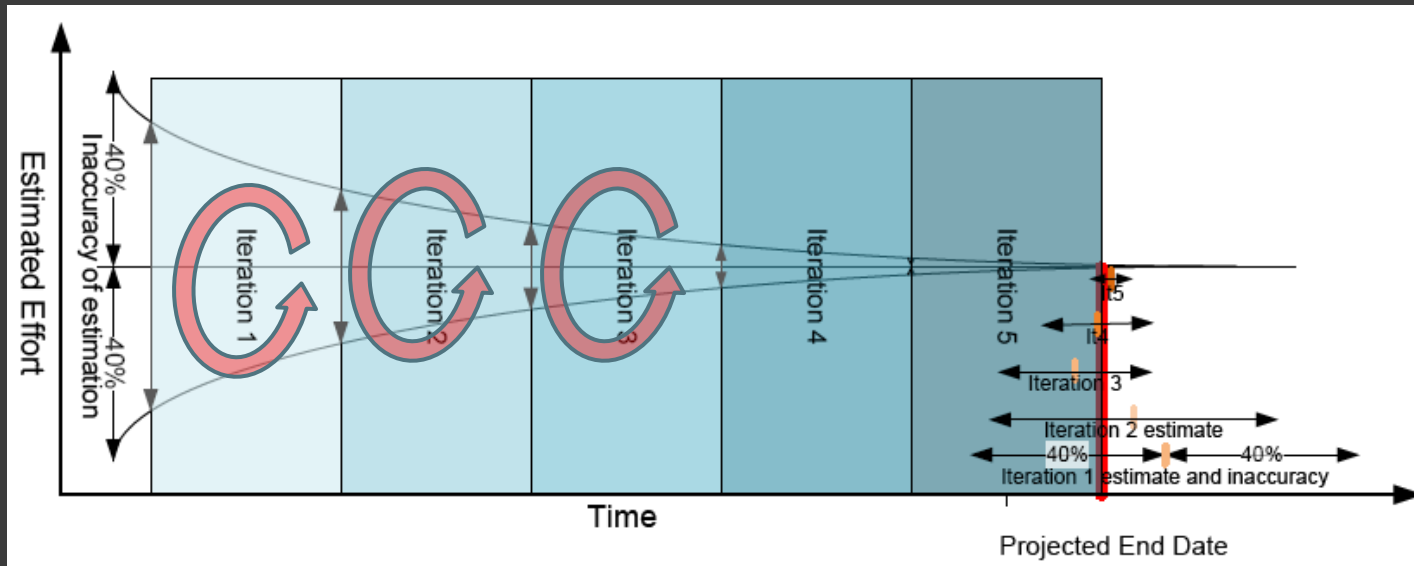
What Does An Agile Process Look Like?



LARGE SCALE PROJECTS



WHAT ABOUT PLANNING?



ITERATIVE GOAL PURSUIT

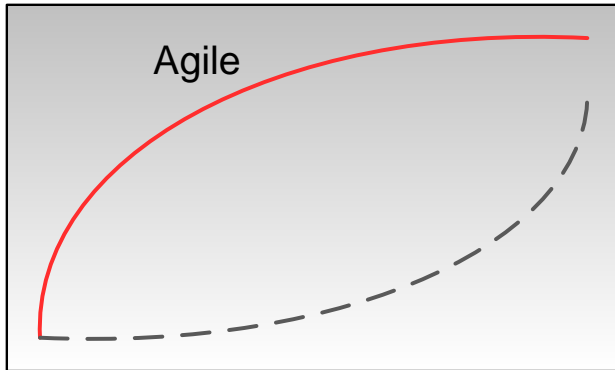


VS.

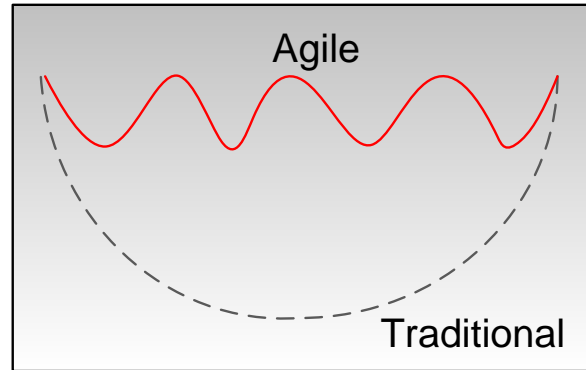


VALUE PROPOSITION

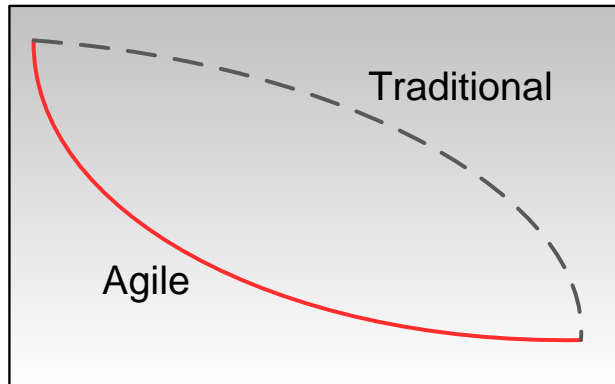
Value



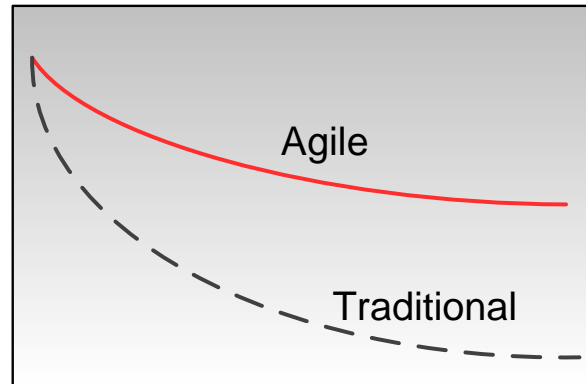
Visibility



Risk



Adaptability



PRACTICAL EXPERIENCE IN THE DEVELOPMENT OF TELEMATICS AND OTHER COMPLEX SOFTWARE SYSTEMS

- My Background
- The challenges of telematics systems development
- Careful, it is not as easy as it seems
- The Power of Agility
- Practical experience in the development of telematics and other complex software systems**
- Conclusion

There is no silver bullet.

-Frederick P. Brooks

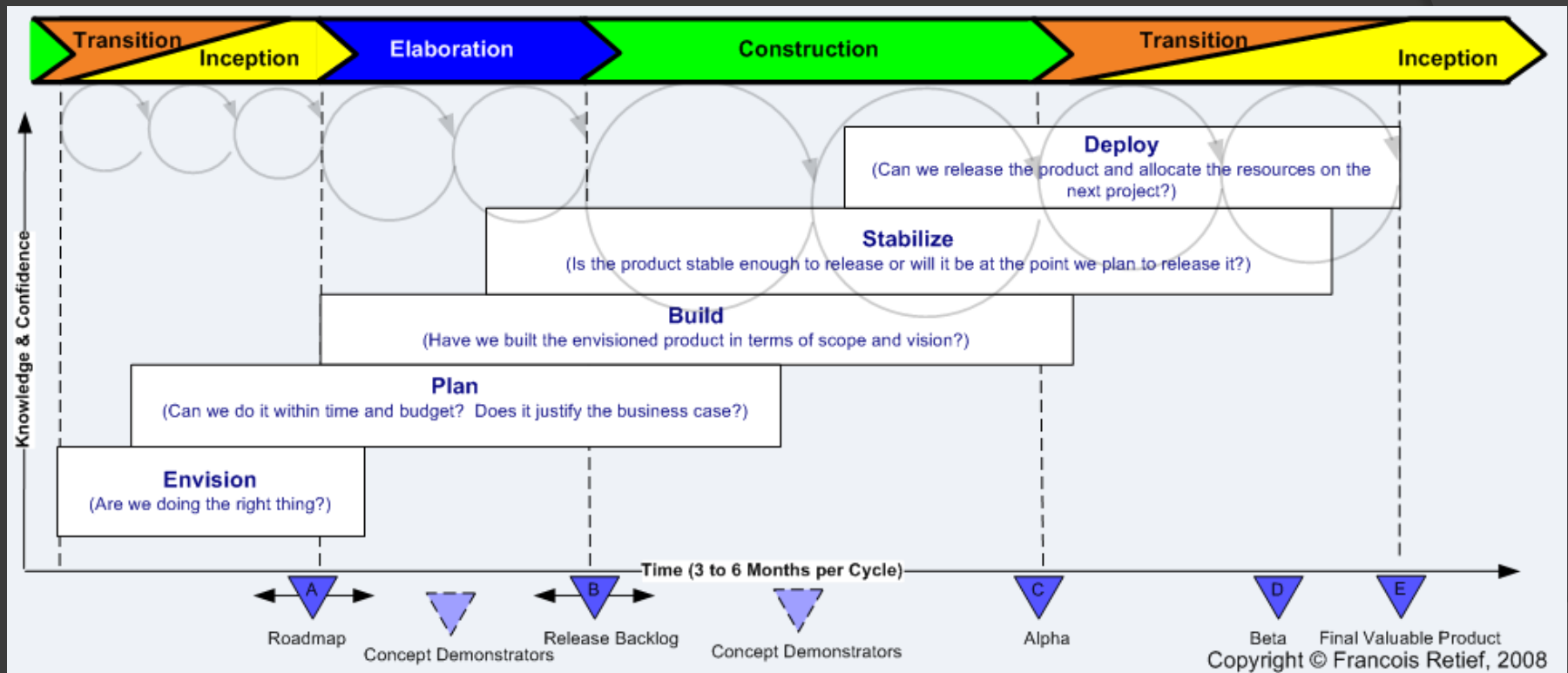
A SHORT PROCESS HISTORY

- ◎ Version 1.0
 - Fast
 - Chaotic
 - Poor Repeatability
 - Lots of Expensive Rework
- ◎ Structured Waterfall Development
 - Repeatable
 - Long Timelines
 - Very complex planning
 - Out of sync development on risk
 - Poor feedback
 - Overlapping phases

A SHORT PROCESS HISTORY

- ◎ A brand new company
 - 9 months to build a company and product
- ◎ Implementation in an existing company
 - Started unreliable
 - Repeatable and Quick
 - Much better visibility
 - Greater stakeholder satisfaction

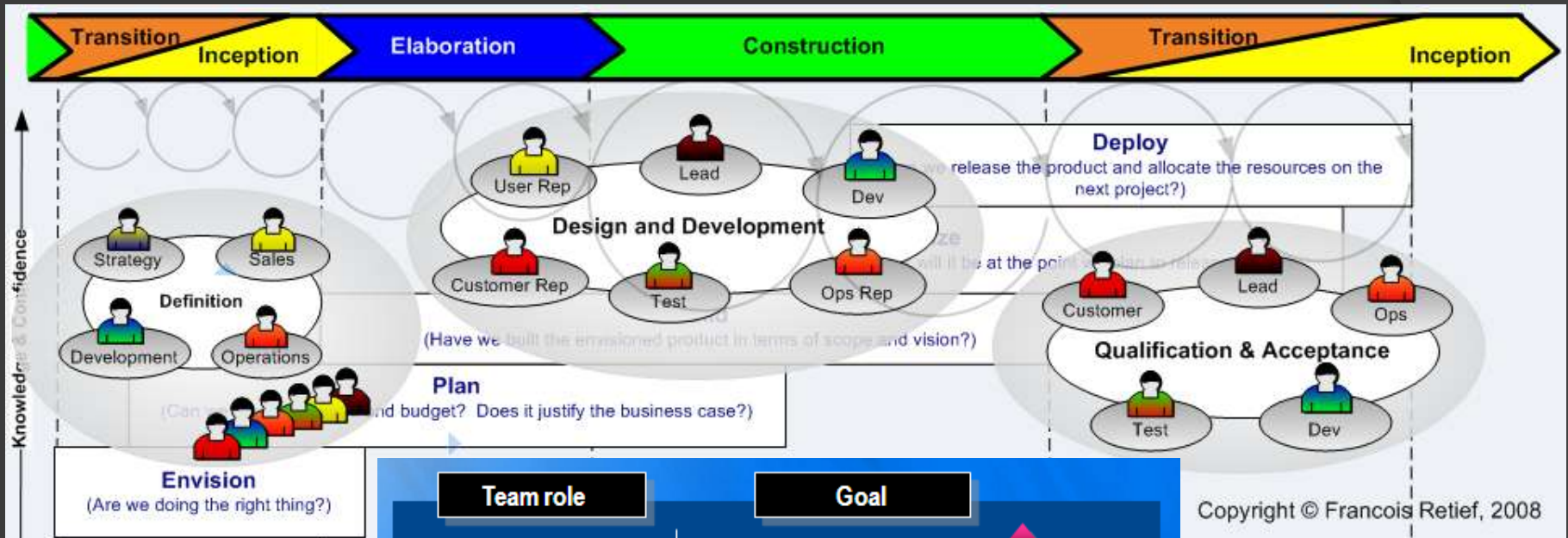
HIGH LEVEL PROCESS OVERVIEW



THINGS TO DO BEFORE YOU START



GET THE RIGHT PEOPLE INVOLVED



Copyright © Francois Retief, 2008

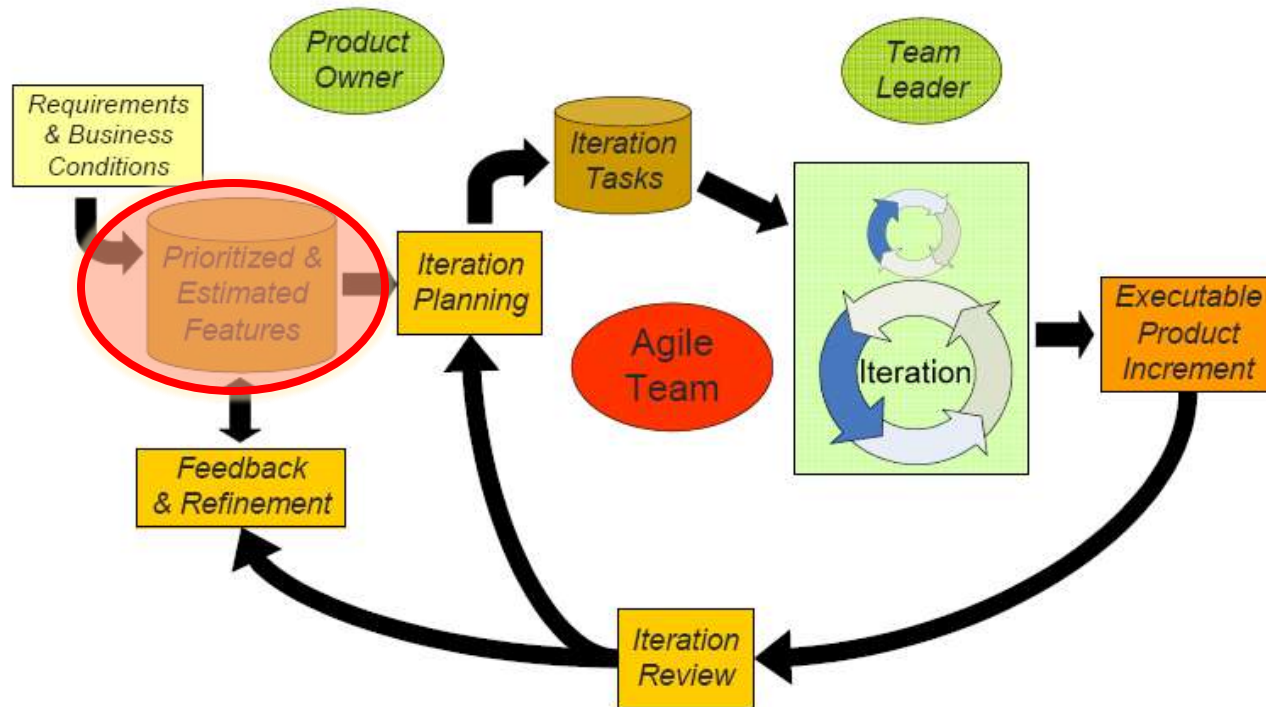
Team role	Goal
Product Management	Satisfied customers
User Experience	Enhanced user effectiveness
Development	Build according to specifications
Test	Release when there are no known issues
Program Management	Delivery within project constraints
Release Management	Smooth deployment / ongoing operations

OTHER THINGS BEFORE YOU START

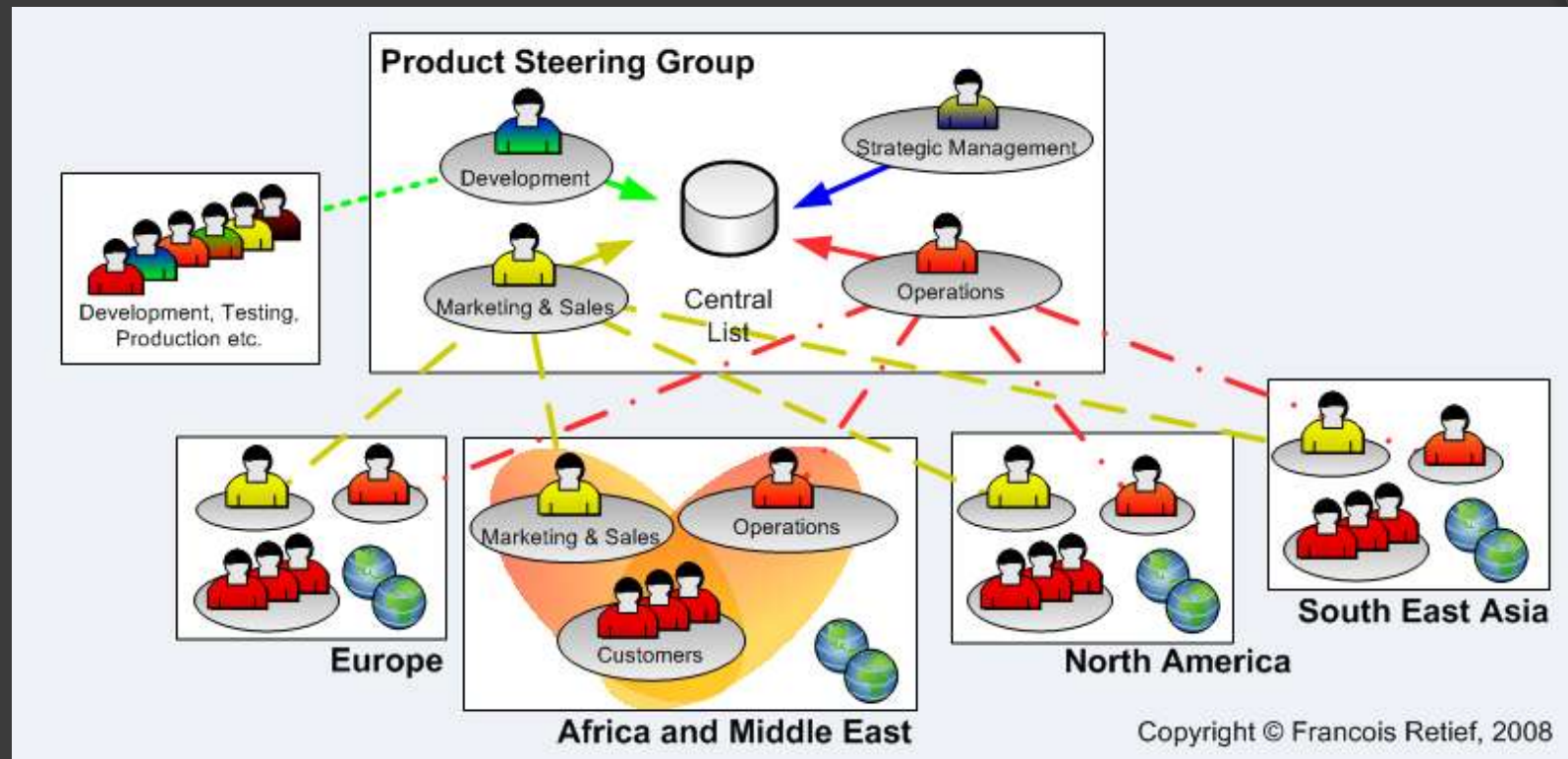
- ◎ Risk Management
 - Target early, drive down actively
 - Mitigation / Contingency
- ◎ Expectation Management
 - Train people unfamiliar with process
 - Set realistic, even conservative goals
- ◎ Change Management
 - Change welcome
 - Hands-off in iteration
 - Baselines
 - Visible change impact

PROCESS OVERVIEW

What Does An Agile Process Look Like?

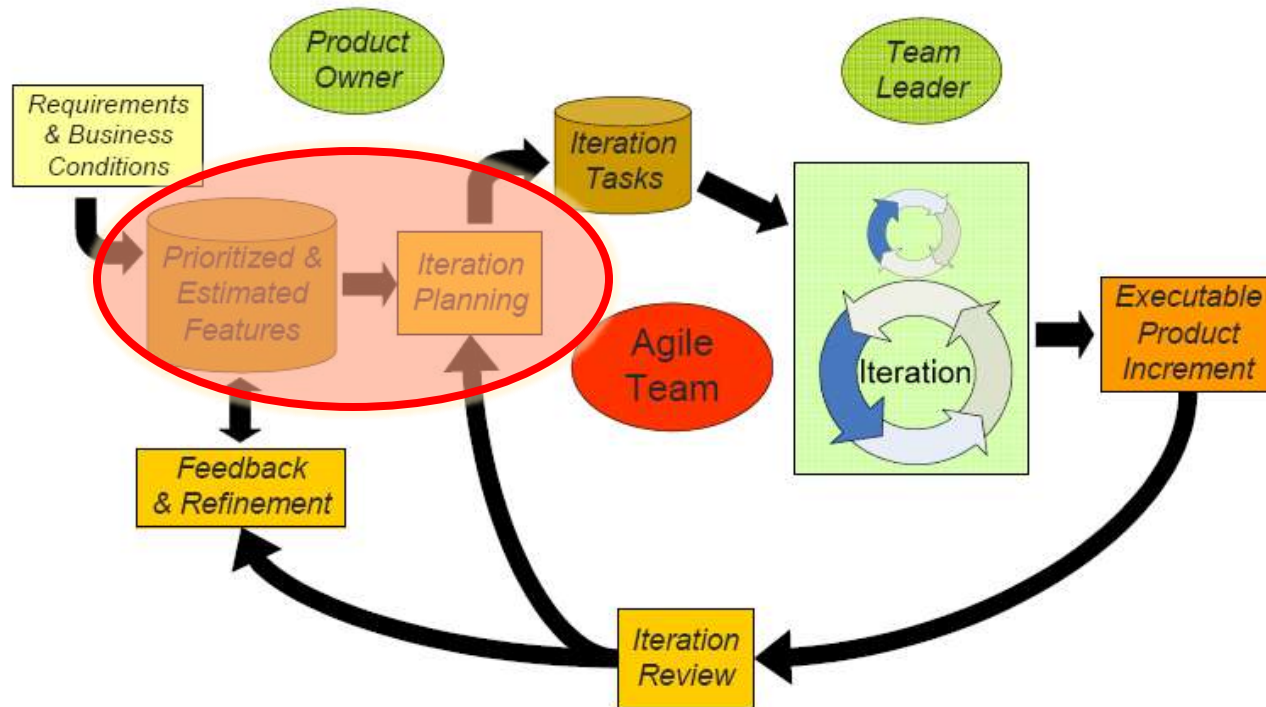


REQUIREMENT GATHERING AND PRIORITIZATION



PROCESS OVERVIEW

What Does An Agile Process Look Like?

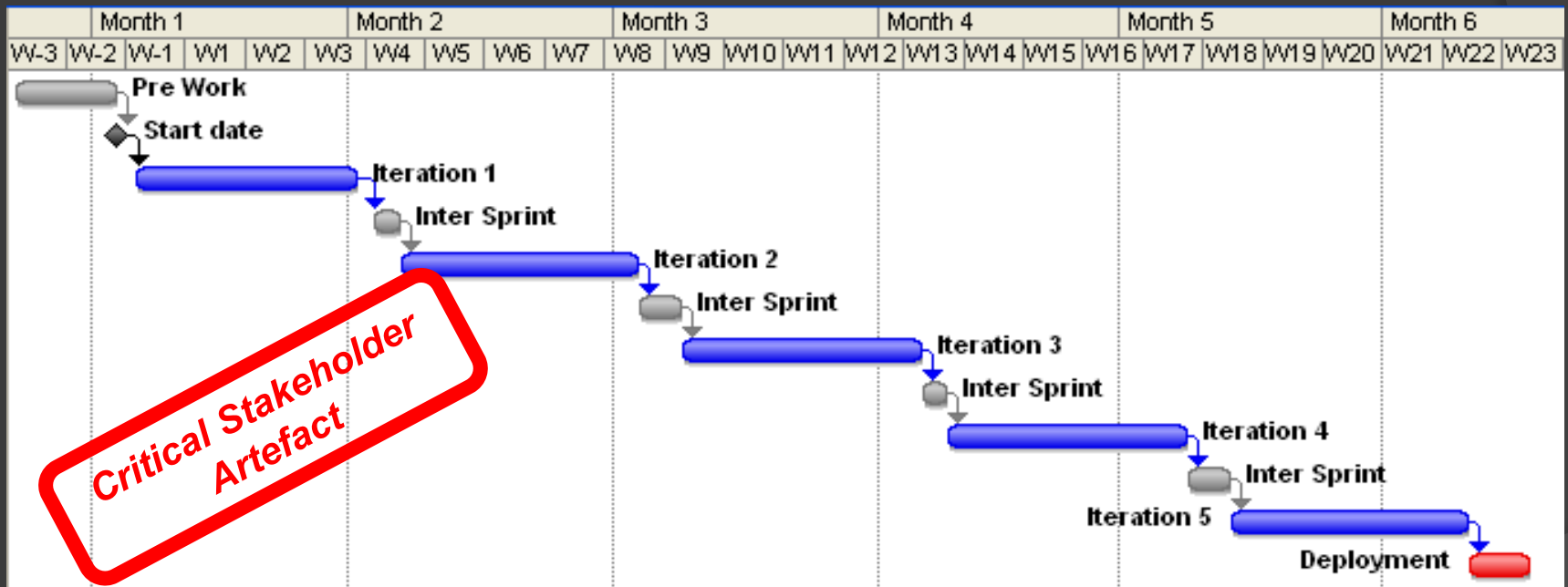


RELEASE AND ITERATION PLANNING

Product Backlog							
Backlog Item	Importance	Risk	Quick Estimate	Priority	Iteration		
Value Item 1	1	2	25	1	1		
Value Item 6	2	2	22	3	1		
Value Item 3	2	3	3	6	1		
Value Item 4a	1	1	30	4	2		
Value Item 2	2	2	18	5	2		
Value Item 4b	1	1	27	4	3		
Value Item 8a	1	3	23	2	3		
Value Item 8b	1	3	19	2	4		
Value Item 5	3	1	25	11	4		
Value Item 11	2	2	5	7	4		
Value Item 10	3	2	2	10	5		
Value Item 9	3	3	1	8	5		
Value Item 7	2	1	34	9	5		
Total			234				

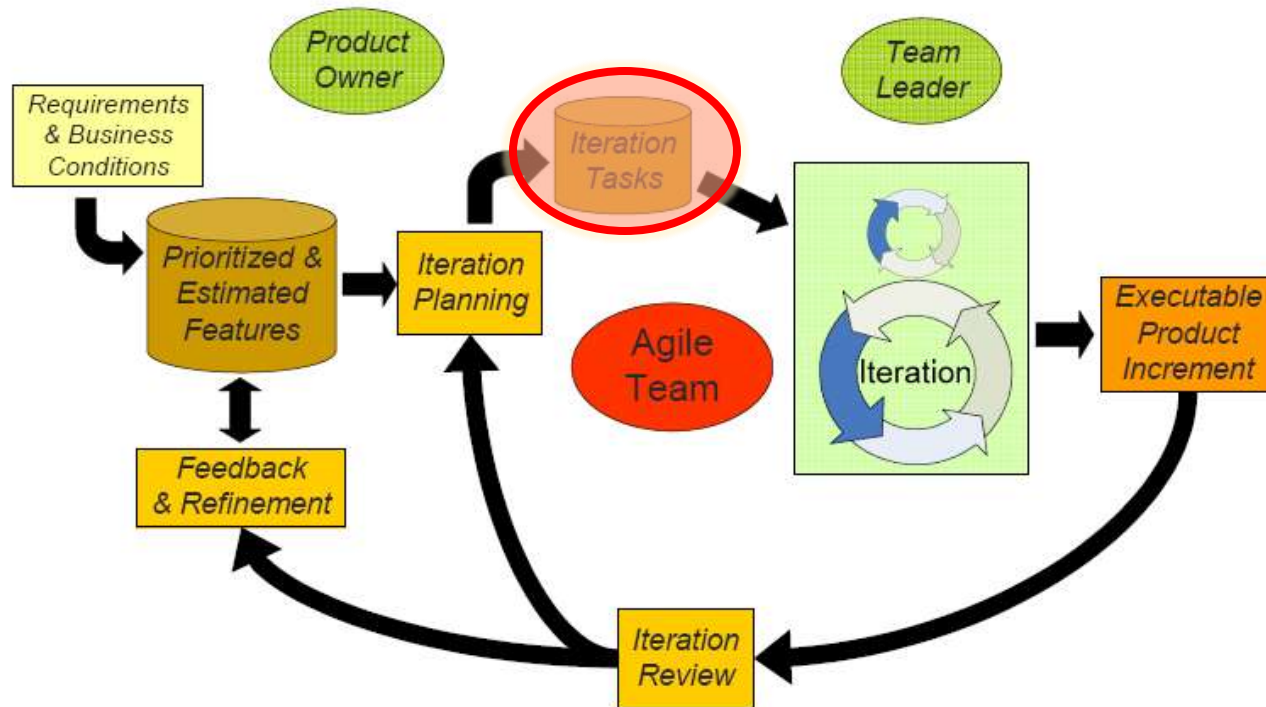
Critical Stakeholder Artefact

RELEASE AND ITERATION PLANNING



PROCESS OVERVIEW

What Does An Agile Process Look Like?



DETAIL ITERATION PLANNING AND PROGRESS TRACKING

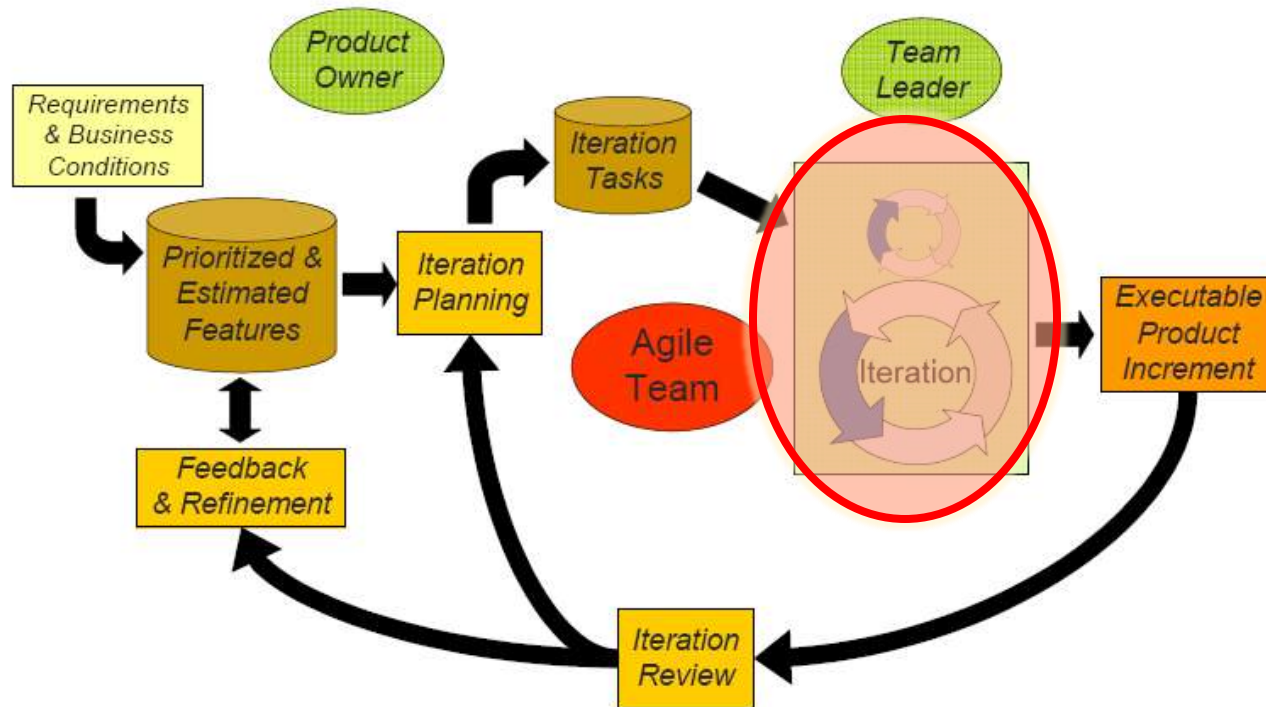
Iteration 1 Detail Planning						
Backlog Item		Risk	Detail Estimate	Priority		
Value Item 1		2	25	1		
Task 1.1 (Acceptance Test)			2	AA		0
Task 1.2 (Acceptance Test)			5	BB		4
Task 1.3			4	BB		2
Task 1.4			3	AA		3
Task 1.5			4	CC		4
Task 1.6			5	CC		5
Task 1.7			2	DD		2
Value Item 6		2	22	3		11
Task 6.1 (Acceptance Test)			5	AA		0
Task 6.2 (Acceptance Test)			3	DD		3
Task 6.3 (Acceptance Test)			4	BB		4
Task 6.4			4	DD		0
Task 6.5			2	DD		1
Task 6.6			1	CC		0
Task 6.7			1	CC		1
Task 6.8			2	DD		2
Value Item 3		3	3	6		0
Task 3.1 (Acceptance Test)			1	CC		0
Task 3.2			2	AA		0
TOTAL			50			31

NB! Task should take between 1 and 3 days to complete

Critical Team Artefact

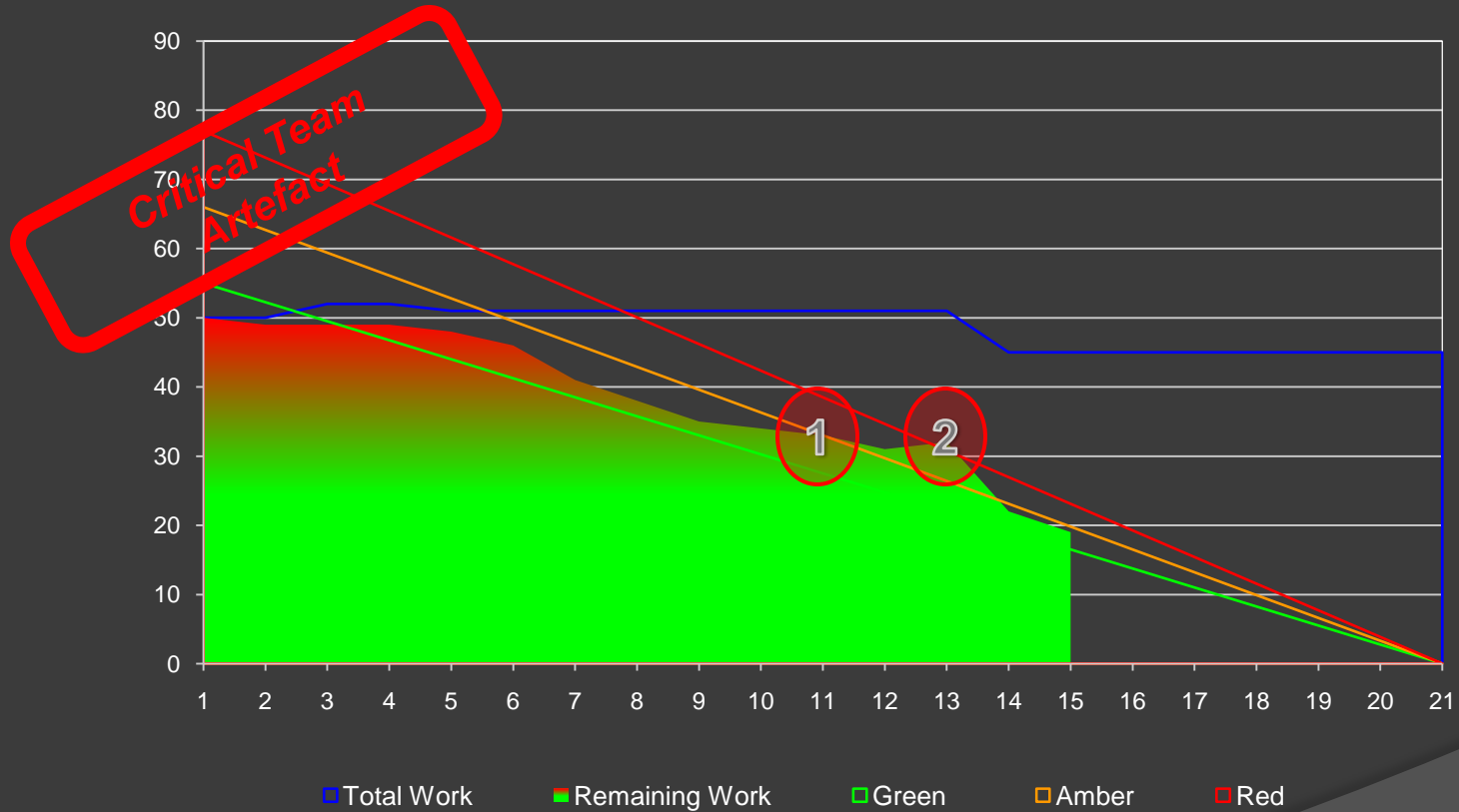
PROCESS OVERVIEW

What Does An Agile Process Look Like?

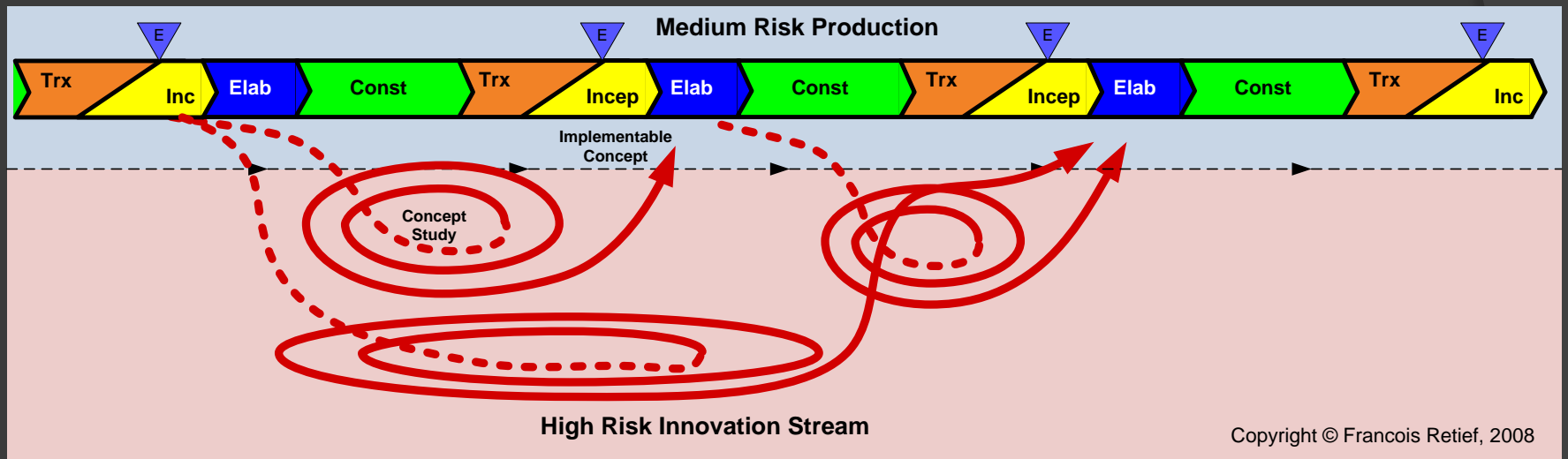


BURN DOWN CHART

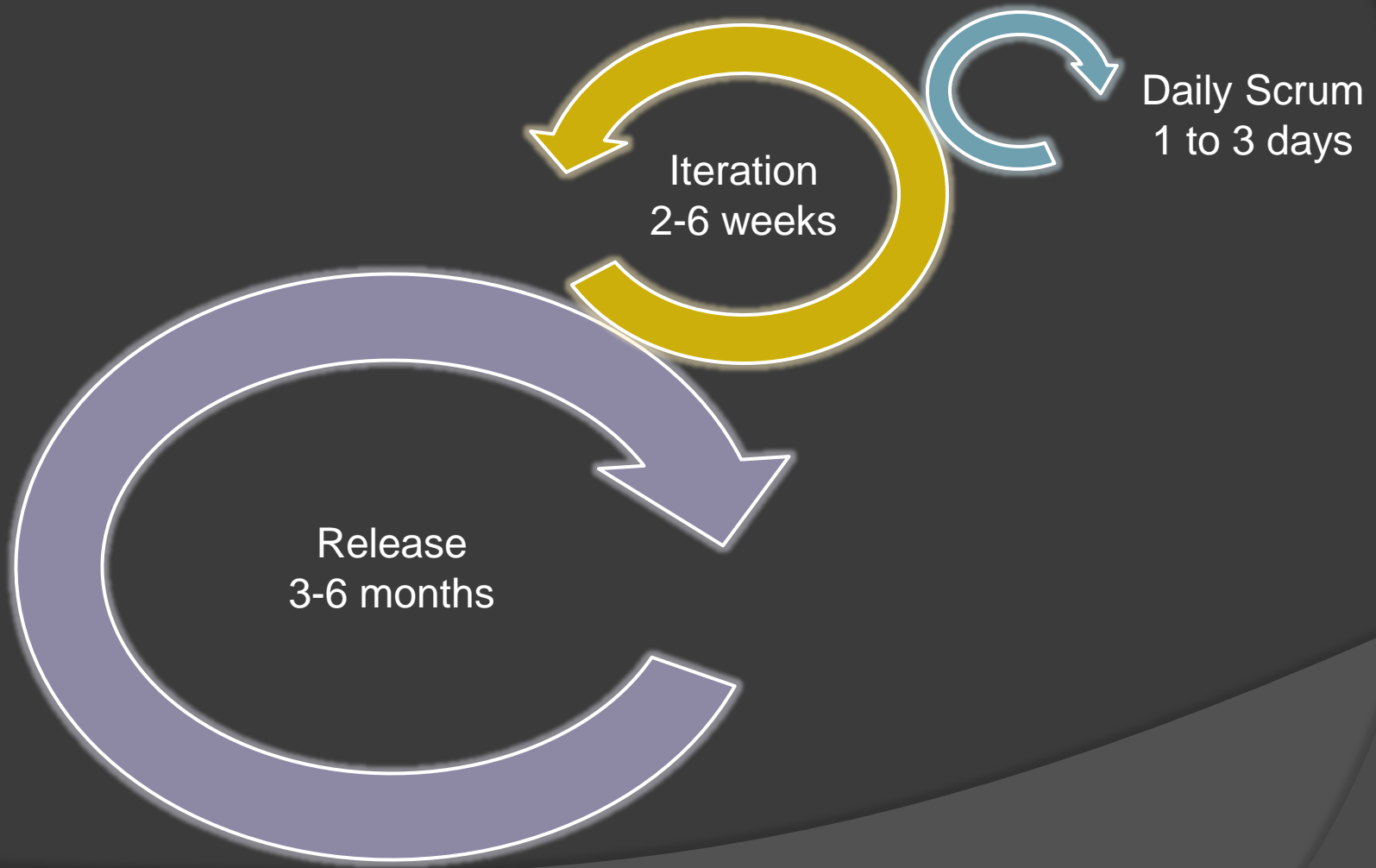
Iteration 5 Burn Down Chart



HOW DOES ONE SCHEDULE INNOVATION?



WHEELS WITHIN WHEELS



DAILY SCRUM MEETING

- ⦿ 15 minute Stand Up meeting: Same Place, Same Time
- ⦿ Have to attend, if by proxy
- ⦿ Critical Artefacts Updated Before Meeting
 - Have I been able to finish what I committed to yesterday?
 - What am I comfortable to commit to have done by tomorrow?
 - What obstacles do I need removed to meet my goals?
 - What obstacles do I foresee I'm going to put in other team members' path?
- ⦿ Take off-topic items offline.

RELEASE BY CONSENSUS

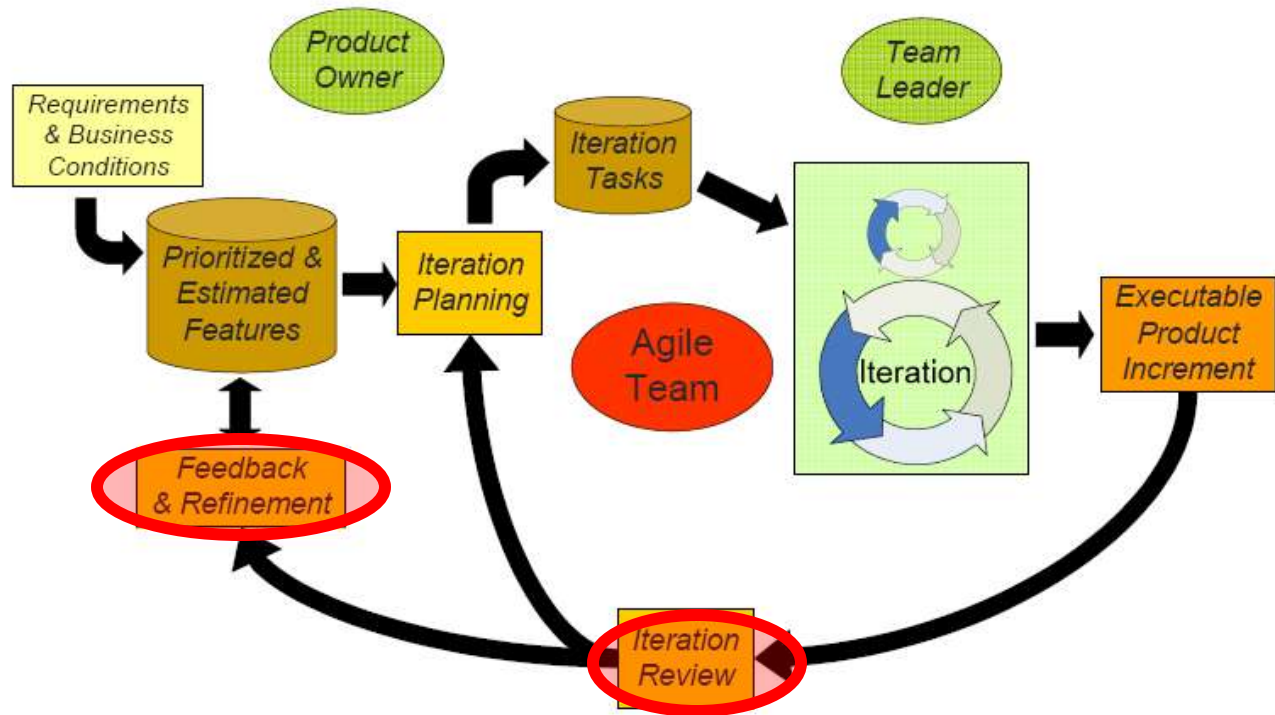


CONTINUOUS IMPROVEMENT



CONTINUOUS IMPROVEMENT

What Does An Agile Process Look Like?



CONCLUSION

- My Background
- The challenges of telematics systems development
- Careful, it is not as easy as it seems
- The Power of Agility
- Practical experience in the development of telematics and other complex software systems
- Conclusion**

Highly beneficial in the Telematics Environment

- ⦿ Improved business value per release
- ⦿ Better visibility
- ⦿ More manageable environment
- ⦿ Improved estimation of delivery
- ⦿ More stakeholder commitment, motivation and satisfaction
- ⦿ Less stakeholder stress

ADAPTATION

- More formal requirement gathering and release management process.
- Despite continuous testing: formal feature freeze period with systems testing. Also large scale field trials and managed deployment process before a final release
- Hardware development via Waterfall methodology with milestones linked into the software/firmware program

SUMMARY & QUESTIONS

- 1) Agile, Iterative Development
- 2) Capable, self-organizing cross functional teams
- 3) Drive risk down continuously
- 4) Separate innovation stream
- 5) Daily stand-up meeting
- 6) Release by consensus
- 7) Retrospectives

