



# SSW-555: Agile Methods for Software Development

## *Crystal*

Dr. Richard Ens  
Software Engineering  
School of Systems and Enterprises





# Acknowledgements

Some of the material in these slides is from *Agile Software Development* by Alistair Cockburn

Some of the material in these slides is from *Crystal Clear: A Human-Powered Methodology for Small Teams* by Alistair Cockburn

# Today's topics

Crystal Clear

Properties

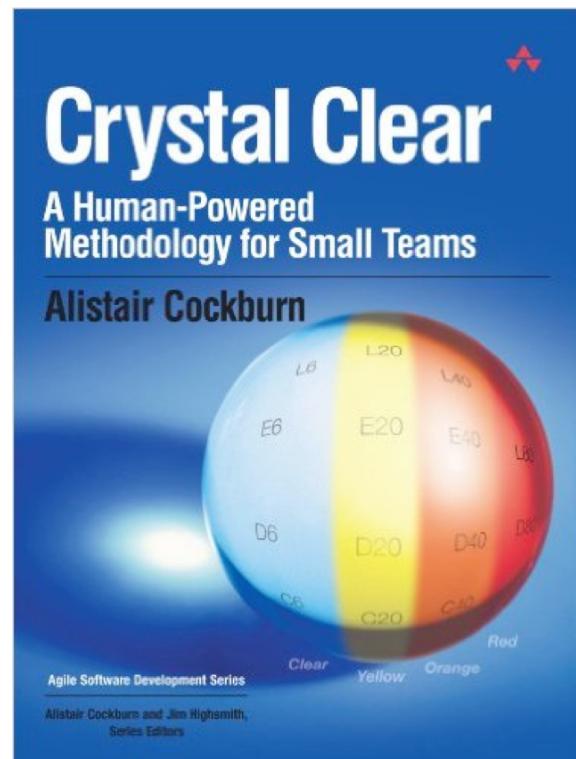
Strategies

Techniques

Roles and work products

Crystal Clear vs. XP

Crystal Family of Methods



# Alistair Cockburn on process and plans

*“Focusing on skills, communications, and community allows the project to be more effective and more agile than focusing on processes and plans.”*

– Alistair Cockburn



Process and planning are important, but people, interactions, community, skills, and talent are more important

Many teams follow the letter of the process rather than the spirit of the process

# Crystal (1992)

Alistair Cockburn interviewed successful projects and extracted the processes that helped lead to success

Teams were succeeding without traditional formal methods

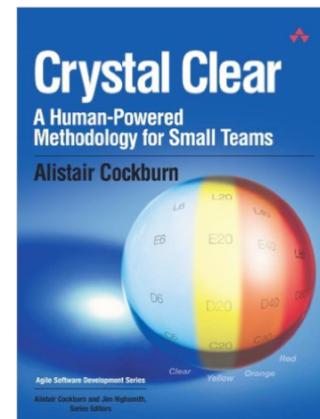
Crystal is a **family** of methods and a set of guidelines that you adjust to meet your project's needs

Different flavors of Crystal for different project needs

Just as no two crystals are the same,  
no two projects are the same

E.g. small projects vs. large projects

Tailor the process to the project



# Crystal Clear

Crystal Clear is the lightest member of Crystal family

Similar to eXtreme Programming

Intended for small, collocated teams (1-6 members)

		Crystal Methodologies				
		Clear	Yellow	Orange	Red	Maroon
Criticality of the Project	Life (L)	L6	L20	L40	L80	L200
	Essential Money (E)	E6	E20	E40	E80	E200
	Discretionary Money (D)	D6	D20	D40	D80	D200
	Comfort (C)	C6	C20	C40	C80	C200
		1 to 6	7 to 20	21 to 40	41 to 80	81 to 200
Number of People involved in the Project						

Source: Crystal Clear: A Human-Powered Methodology for Small Teams



# Properties, Strategies, Techniques

# Properties

# Properties

# Properties should be true about any project

e.g. Principles in the Agile Manifesto

# Strategies

# Strategies are a plan to accomplish a goal

# Higher level approaches to solving problems

# Techniques

# Techniques are things you do

**Skills that the team can develop and use to solve problems**





# Properties across all Crystal Methods

## 1. Frequent delivery (Clear)

Intervals of 2 weeks to 4 months

Tune Crystal parameters to your project's needs

## 2. Reflective improvement (Clear)

“Reflection workshop” every few weeks, e.g. after every delivery

Review what parts of process worked and what needs to change

Analogous to Sprint Retrospective

Properties

# Properties across all Crystal Methods

## 3. Osmotic communication (Clear)

See, hear, and absorb  
information in background

Improves communication  
across team

All team members are aware of  
others' work and can help  
or take over if needed

Goal: communications and  
community

Properties





# Properties across all Crystal Methods

## 4. Personal safety

### Properties

Everyone should be comfortable sharing ideas with everyone  
Mutual trust across the team, including the “boss”  
Speak up without fear of ridicule

## 5. Focus

Minimize interruptions from calls, email, meetings

At least two hours per day of uninterrupted time

Clear direction and priorities for the project

Everyone knows what to work on

People have time and peace of mind to work



# Properties across all Crystal Methods

## 6. Easy access to domain experts

Properties

Onsite or via regular meetings

Analogous to onsite experts from eXtreme Programming

## 7. Technical environment

Automated tests

Configuration management

Frequent/Continuous integration

## 8. Collaboration across organizational boundaries

# Strategies

## I. Exploratory 360°

Planning at the beginning of the project

Early requirements and domain model

Demonstrations of feasibility

## 2. Early victory

Focus on an easy first deliverable

Winning helps the team to bond and feel successful



# Strategies

## 3. Walking skeleton

Tiny end-to-end version of the system's functionality

Provides a simple model for the user to explore

System evolves from this first architecture



## 4. Incremental re-architecture

Be prepared to change the architecture



Have a fallback or temporary solution (a short term hack)

Evolve to the new architecture through parallel effort

# Strategies

## 5. Information radiators

Display useful information through osmotic communication

Updated continuously

E.g. Burn down chart to display status and progress



**How are  
we doing?**



# Techniques

# I. Methodology shaping to define the process



Conduct interviews to collect information about the team

**Conduct workshop to define an optimal process for this team**

## 2. Reflection workshop

**Hour-long meeting after each delivery**

# What are we doing well?

# What can we do better?

# How can we improve our process?

# Analogous to Sprint Retrospective

<p><b>Keep these</b></p> <ul style="list-style-type: none"><li>test lock-down</li><li>quiet time</li><li>daily meetings</li></ul>	<p><b>Try these</b></p> <ul style="list-style-type: none"><li>pair testing</li><li>fines for interruptions</li><li>programmers help testers</li></ul>
<p><b>Problems</b></p> <ul style="list-style-type: none"><li>too many interruptions</li><li>shipping buggy code</li></ul>	

Source: Crystal Clear: A Human-Powered Methodology for Small Teams



# Techniques

### 3. Blitz planning/Planning “Jam Session”

# Like XP's Planning Game

## Create initial project plan

# Include all stakeholders

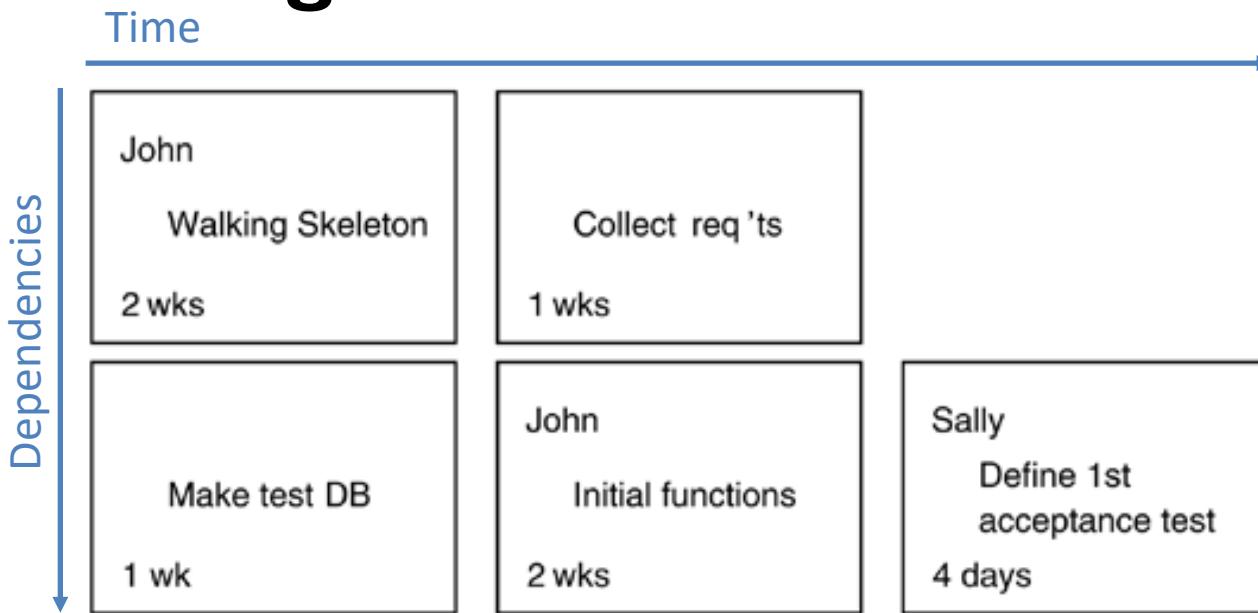
# Executive sponsor, users, developers

# Planning Game vs Blitz Planning



Planning Game	Blitz Planning
User Stories	Tasks
Assume no dependencies between stories	Explicitly analyze dependencies between tasks
Fixed length sprints	Variable iteration durations

# Blitz Planning Cards



- Identify tasks
- May identify task owner
- Identify task duration
- Identify task order and dependencies

# Techniques

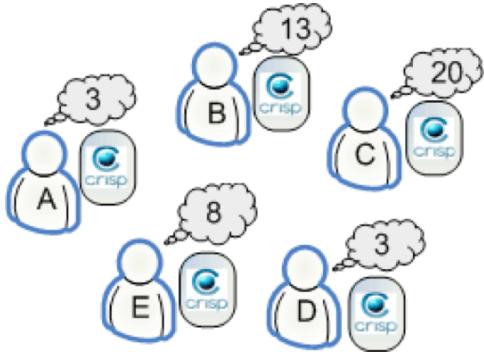


## 4. Delphi estimation

Iterative estimation by groups of experts

Achieve convergence through peer pressure

Planning Poker



## 5. Daily stand-ups

Borrowed from Scrum

What did I accomplish since last time?

What's my goal for today?

What problems do I have?



# Techniques

## 6. Agile interaction design

# User-centered design

# What would the user want?

# How will the user use the system?

## Focus on roles of users

# Human Computer Interaction (CS 545)

# Usability testing



# Role modeling session



Each card represents a feature or user story

Specify the goal on the card

Work through a scenario of a user accomplishing a task

# Techniques

## 7. Process miniature

Train participants on processes with short exercises

Peter Merel's “Extreme Hour” to learn XP

Walk through all aspects of XP on simple project in 60 minutes



# Techniques

## 8. Side-by-side programming

Sit close enough to see one another's screens

Not quite pair programming:  
Help is available when needed,  
but not constant

Osmotic communication



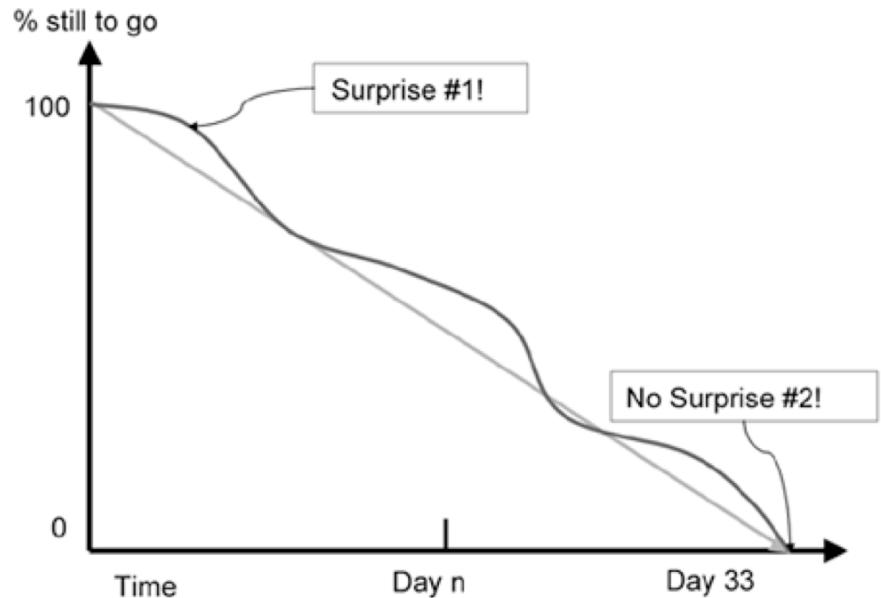
# Techniques



## 9. Burn charts

Display progress against goals

Borrowed from Scrum





# Roles and Work Products

Smaller projects require fewer roles

## Executive Sponsor

Mission statement

## Team

Team structure and conventions

Reflection workshop results

## Coordinator/Manager

Project map

Release plan

Project status

Risk list

Iteration plan and status

Viewing schedule



# Roles and Work Products

Larger projects require additional roles

**Business expert and  
expert user**

Actor-goal list

Use cases and  
requirements file

User role model

**Lead designer**

Architecture description

**Tester**

Bug reports

**Designer-programmer**

Screen drafts

Common domain model

Design sketches and notes

Source code

Migration code

Tests

Packaged system

**Writer**

User help text

# Crystal Clear vs. eXtreme Programming

Many similarities, but...

Cockburn claims that:

XP requires more discipline than Crystal Clear

XP has 12 mandatory practices, e.g. Pair Programming, TDD, ...

XP can be more productive than Crystal Clear

It is easier to start using Crystal Clear

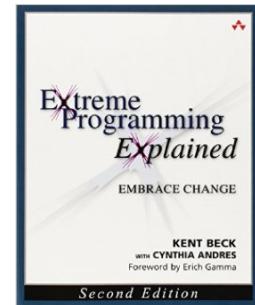
Especially for Plan Driven teams

It is easy to fall back from XP to Crystal Clear

Use only the XP practices that work for your project

		Crystal Methodologies				
		Clear	Yellow	Orange	Red	Maroon
Criticality of the Project	Life (L)	L6	L20	L40	L80	L200
	Essential Money (E)	E6	E20	E40	E80	E200
	Discretionary Money (D)	D6	D20	D40	D80	D200
	Comfort (C)	C6	C20	C40	C80	C200
		1 to 6	7 to 20	21 to 40	41 to 80	81 to 200
Number of People involved in the Project						

vs





# The family of Crystal Methods

A single process is not likely to work for all projects

Adapt your method according to:

Size in staff

I – 6 (Crystal Clear)

7 – 20 (Crystal Yellow)

21 – 40 (Crystal Orange)

41 – 80 (Crystal Red)

Potential risks for causing damage

Comfort (C)

Discretionary money (D) – the company will survive if the project fails

Essential money (E) – the company may fail if the project fails

Life (L) – safety critical

Note that Life (safety critical projects) should NOT use Crystal Methods

Source: <http://www.devx.com/architect/Article/32836/0/page/2>

		Crystal Methodologies				
		Clear	Yellow	Orange	Red	Maroon
Criticality of the Project	Life (L)	L6	L20	L40	L80	L200
	Essential Money (E)	E6	E20	E40	E80	E200
	Discretionary Money (D)	D6	D20	D40	D80	D200
	Comfort (C)	C6	C20	C40	C80	C200
		1 to 6	7 to 20	21 to 40	41 to 80	81 to 200
Number of People involved in the Project						



# Crystal Orange

## Roles

Sponsor  
Business expert  
Usage expert  
Technical facilitator  
Business analyst/designer  
Project manager  
Architect  
Design mentor  
Lead designer  
UI designer  
Reuse point  
Writer  
Tester

## Work Products

Requirements document  
Release sequence  
Schedule  
Status reports  
UI design document  
Common object model  
Inter-team specifications  
User manual  
Source code  
Test cases  
Migration code



# Crystal Orange teams and sub-teams

System planning

Project Monitoring

Architecture

Technology

Functions (divided into cross-functional groups)

Infrastructure

External test

# Questions?

