TIMG 5006
Management of Software Engineering Projects

Session 2: Sep 9

Fall 2015

Michael Weiss

www.carleton.ca/tim
www.carleton.ca/tim/tim.pdf

Session 2 objectives

- Upon completion of this session, you will know about
  - Challenges in software project management
- And you will be able to
  - Identify which project management challenges a given approach addresses

Agenda

1. Assignments
2. Adrenaline junkies
3. Management challenges
4. Questions

Readings for Session 2

- Sauer, C., & Reich, B. (2008), Rethinking IT project management: evidence of a new mindset and its implications, International Journal of Project Management, 27, 182-193.
- Baskerville, R., Ramesh, B., Levine, L., Pries-Heje, J., & Slaughter, S. (2003), Is Internet-speed software development different?, IEEE Software, 20(6), 70-77.
- Royce, W. (2005), Successful software development style: steering and balance, IEEE Software, 22(5), 40-47.
- Noll, J., Beecham, S., & Richardson, I. (2010), Global software development and collaboration: barriers and solutions, ACM Inroads, 1(3), 66-78.

1. Assignments

- Write a project management pattern (20%)
- Design a card / board game for training a software project management skill (40%)

Assignment 1 (20%)

- Write a software project management pattern
  - Identify a **problem** and its **context**
  - Discuss what makes it challenging (ie **forces**)
  - Present a **solution** to the problem based on your personal experience or on the literature
  - Discuss **consequences** of the solution
  - List **known uses** of the solution as evidence
- Workshop on **Oct 21** where you will receive feedback on the pattern from the class and final presentation
- Deliverables: topic by **Sep 23**, final version on **Nov 25**

Example (summary)

- Problem: Ensure that subsystems of a larger system developed in an iterative manner work together
- Forces: Subsystems are developed at different rates, and developers work on a private copy of the system when developing their subsystems
- Solution: Give developers a mechanism to integrate software periodically, impose policies that discourage developers from developing without integration
- Consequences: Developers all see the same system
- Known uses: Describe known ways of implementing this solution (eg continuous integration)

Assignment 2 (40%)

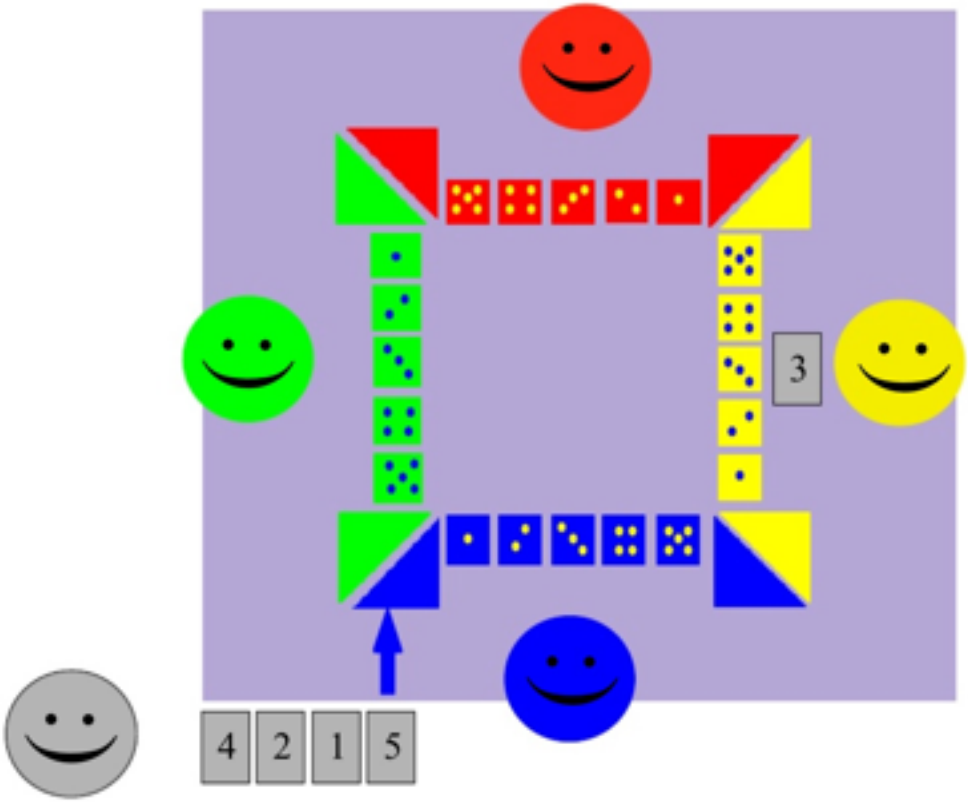
- Design a card / board game for training a software project management skill
- Pitch of game idea on **Sep 30**
- Presentation of first version on **Nov 4**
- Final presentation on **Dec 2**

Elements of a game

- Contents
- Objective
- Game set up
- Playing the game
- Game end
- Glossary

Example (Kanban 1s)

- Kanban 1s game (http://jonjagger.blogspot.ca/2012/02/my-kanban-1s-board-game_19.html)



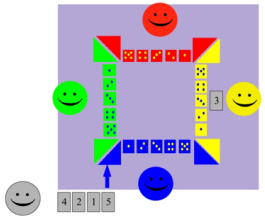
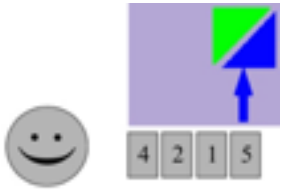
Introduction to Kanban

- Method of product development that emphasizes continuous flow of work through the system
- Based on principles of visualizing workflow, limiting work in progress (WIP), and enhancing cycle time

Backlog	In Progress	Done
Story X	Story P	Story S
Story Y	Story Q	
	Story R	

Kanban
=
"Billboard"

Example (Kanban 1s)

Contents 	4 players & 1 customer (product owner) 4 edges of 5 squares (for stories of different sizes) Story cards (with values 1, 2, 3, 4, or 5) 6 dice
Objective	Teach principles of Kanban Build intuition for work in progress (WIP) principle
Game set up 	Customer creates backlog of story cards Stories enter at one edge

Example (Kanban 1s)

Playing the game 	Each simulated day each player throws 6 dice Each 1 counts as a unit of work Move one story one square per unit of work When story reaches square matching size it moves to the done corner of the edge Next player picks up story from corner of its edge
Game end	Stories reach the done stage of the last player (or, alternatively, time runs out)
Glossary	Story … Backlog … Edge … …

Websites

- An excellent resource with many agile games: http://tastycupcakes.org
- List of "lean" games: http://www.leansimulations.org
- Serious games for product development: http://www.innovationgames.com

2. Adrenaline Junkie

System Development Lemming Cycle	
Seasons for Change	
Paper Mill	
Shipping on Time, Every Time	
Projects with Rhythm	

Patterns (good, bad, and ugly)

System Development Lemming Cycle	The project team slavishly adheres to an un-tailored process standard.
Seasons for Change	Your tolerance for change must diminish with life of the project.
Paper Mill	Some organizations measure progress by the number of documents produced.
Shipping on Time, Every Time	The team always ships on time.
Projects with Rhythm	The team establishes a rhythm for its work by delivering at regular intervals.

3. Management challenges

- Need to rethink project management to include value creation, social, and knowledge processes
- Developing software at high speed
- Steering leadership is better than plan-and-track
- Distributed project teams

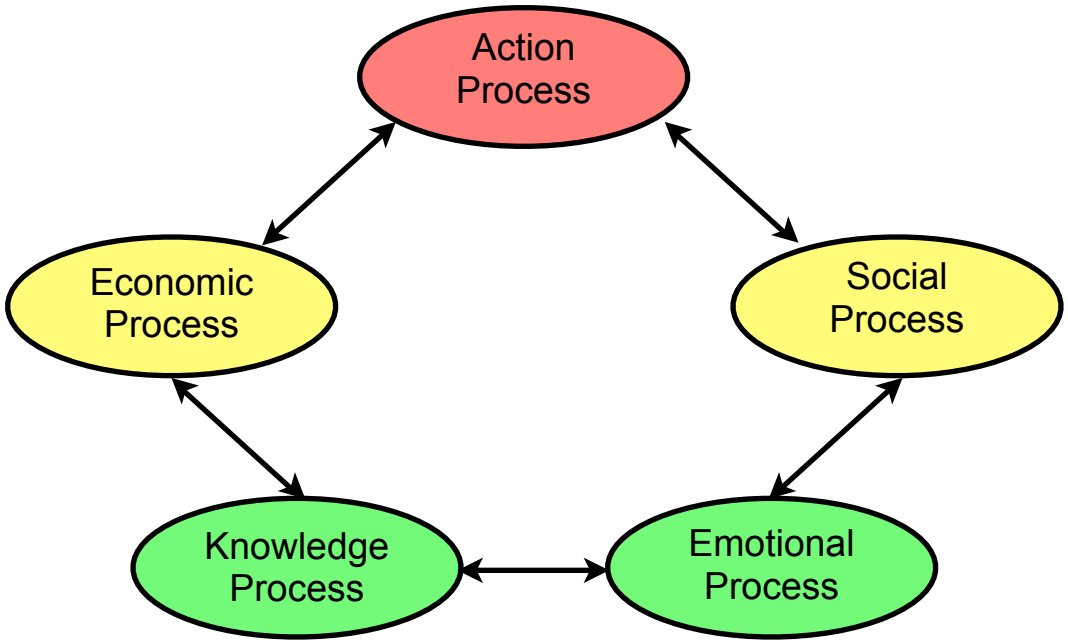
Exercise: Vacation planning

- How would you plan a vacation?

Rethinking project management

- Recent study on rethinking project management
- Projects can be conceptualized in terms of three processes: **action** process, **economic** process (what value is created), and **social** process (human)
- More broadly, project management also includes a **knowledge** process (reflection), and an **emotional** process (personal identification, trust)

Key processes



Principles of new mindset

- Focus on ultimate **value** (value creation)
- *Identification* with **goals** (value creation, *emotional*)
- Investment in **trust** (social process)
- Collective **responsibility** (response to complexity)
- Willingness to **adapt** (value creation)
- **People** development (reflection)
- **Learning** orientation (reflection)
- **Innovation** and creativity (value creation, reflection)
- **Proactive** view (value creation, *emotional*)

Insights

- Identifies **key processes** of a new mindset for software project management
- Describes **principles** that organizations and project managers can use to take concrete action
- People development, learning orientation (reflection), and innovation/creativity relate to **knowledge**
- Projects create **emotional** stress, which can be managed through personal identification with project goals and a proactive view grounded in values

Reducing cycle time

- Traditional software engineering methods assume stable and disciplined processes
- Internet environment intensifies SW development problems by focusing on **shorter cycle times**
- New **market environment** (ie providers cannot rely on customers to define expectations)
- Lack of experience developing for new **technology**

High-speed development practices

- Develop releases in **parallel** (overlap)
- Release **more often** to cope with **fluid** requirements
- Depend on **tools** that speed up development
- Implant customers in development environment to gain direct and constant **access to customers**
- Establish **stable architecture** standardized across multiple projects (eg Model-View-Controller)
- Assemble third-party or off-the-shelf **components**
- Ignore maintenance (**start from scratch**)!
- Adjust the methodology frequently for **just enough process** to deliver the software on time

Insights

- Cost and quality do not drive Internet-speed software development, but **speed** (!) is most important
- Projects do not have beginning or end, but are ongoing: consider a **series of products**
- Maintenance (releases) is sometimes merged with introduction of new functionality (releases)
- Human resources are less interchangeable in Internet-speed development as **experience matters**

Managing a movie production

Unique to software projects

- Analogy between movie & software production
  - No laws of physics, only IP
  - Quality is evaluated by audience
  - Anything can change
  - No predetermined order
  - Low success rate
- Initially, only ideas and $ constrain project
- Quality is subjective: best measured by customers
- Requirements are negotiable

Steering leadership

- Management of software projects needs to reconcile the uncertainties of **problem space** (user needs) with those of the **solution space** (architecture, technology) and the **planning space** (cost, time, resources, capabilities) mapping between the other spaces
- Iterative approaches allow **adjustments** to be made to choices in each of those spaces
- **Uncertain nature** must be reflected in the precision of project management artefacts (eg requirements)
- Put another way, early precision is **only a facade**

Patterns for steering leadership

- **Scope** management (solutions and user requirements evolve by mutual adjustment)
- Process **rigor** (manage the creative process by being light on rigor initially, and increasing it with time)
- Process **honesty** (deliver a sequence of intermediate results, some of which will be dead ends)
- **Quality** control (make testing a first-class citizen, run integration tests of the whole system early)

Insights

- Iterative approach is based on **results** not activities
- Implementing a steering style of leadership improves **time to value** (ie we will deliver results sooner)
- By integrating as **early** as during design through a series of working releases, integration problems can be addressed early avoiding expensive fixes later
- Doing so is more cost-effective: in traditional projects up to 40% of resources are spent on integration and test due to "scrap and rework"; projects with iterative process and steering leadership consume only 25%

Managing distributed projects

- Many projects today are distributed, that is, team members are distributed in **space** and **time**
- Outsourcing is an important case: off-shoring, open sourcing, inner sourcing and crowd sourcing
- Motivations include: reducing cost, reaching emergent markets, and "last-mile" delivery (customization)
- Key challenge: **collaboration** among the teams (agreeing to objectives, allocating tasks, establishing interfaces, and managing dependencies)

Barriers to distributed projects

- What are the **barriers** to distributed projects?

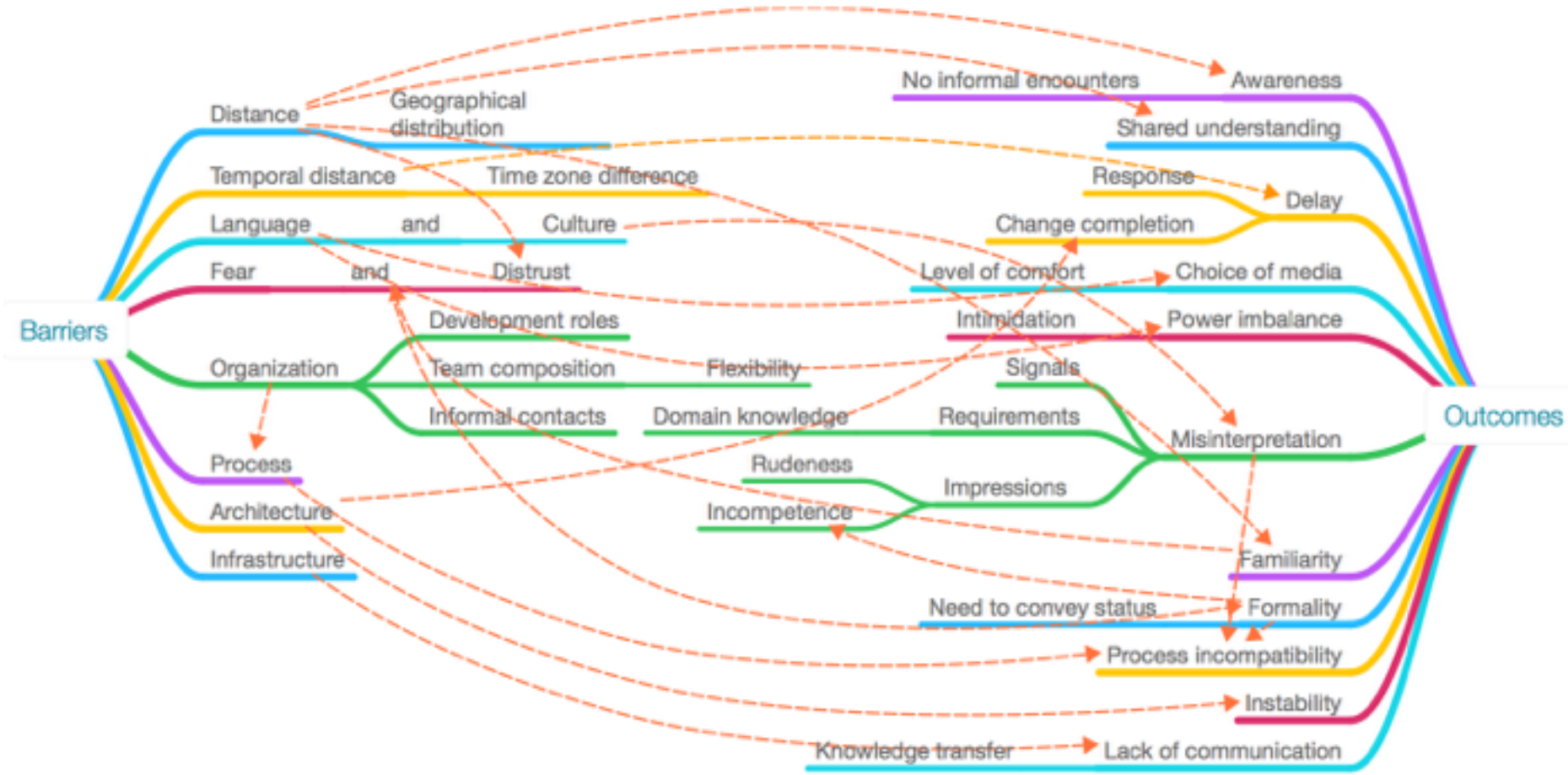
- Work in groups to identify common themes from a list of references on global software development
- Online participants please use references on next page
- In-class participants will receive copies in class

Exercise (online)

- Technology Selection to Improve Global Collaboration
- Overcoming Requirements Engineering Challenges: Lessons from Offshore Outsourcing
- Sysiphus: Enabling informal collaboration in global software development
- Tactical Approaches for Alleviating Distance in Global Software Development
- A Practical Management and Engineering Approach to Offshore Collaboration
- Collaboration Patterns and the Impact of Distance on Awareness in Requirements-Centred Social Networks

Barriers …

Barriers and outcomes



Barriers and outcomes

Barrier	Outcome
Distance	Awareness
Temporal distance	Shared understanding
Language and culture	Delay
Fear and distrust	Choice of media
Organization	Power imbalance
Process	Misinterpretation
Architecture	Familiarity
Infrastructure	Formality
	Process incompatibility
	Instability
	Lack of communication

Questions

- What are the top three management challenges?

Readings for next session

- Leonard, D. & Rayport, J. (1997), Spark innovation through empathic design, Harvard Business Review, 75, Nov-Dec, 102-115
- Lehtola, L., Kauppinen, M., Vähäniitty, J., & Komssi, M. (2009), Linking business and requirements engineering: is solution planning a missing activity in software product companies?, Requirements Engineering, 14(2), 113-128
- Weiss (2012a), User frustrations as opportunities, TIM Review, April, http://timreview.ca/article/546
- Weiss (2012b), Creating Customer Value Propositions for Technology Products, EuroPLoP

Credits

- Movie crew, http://www.disneysub.com/files/set1.jpg

