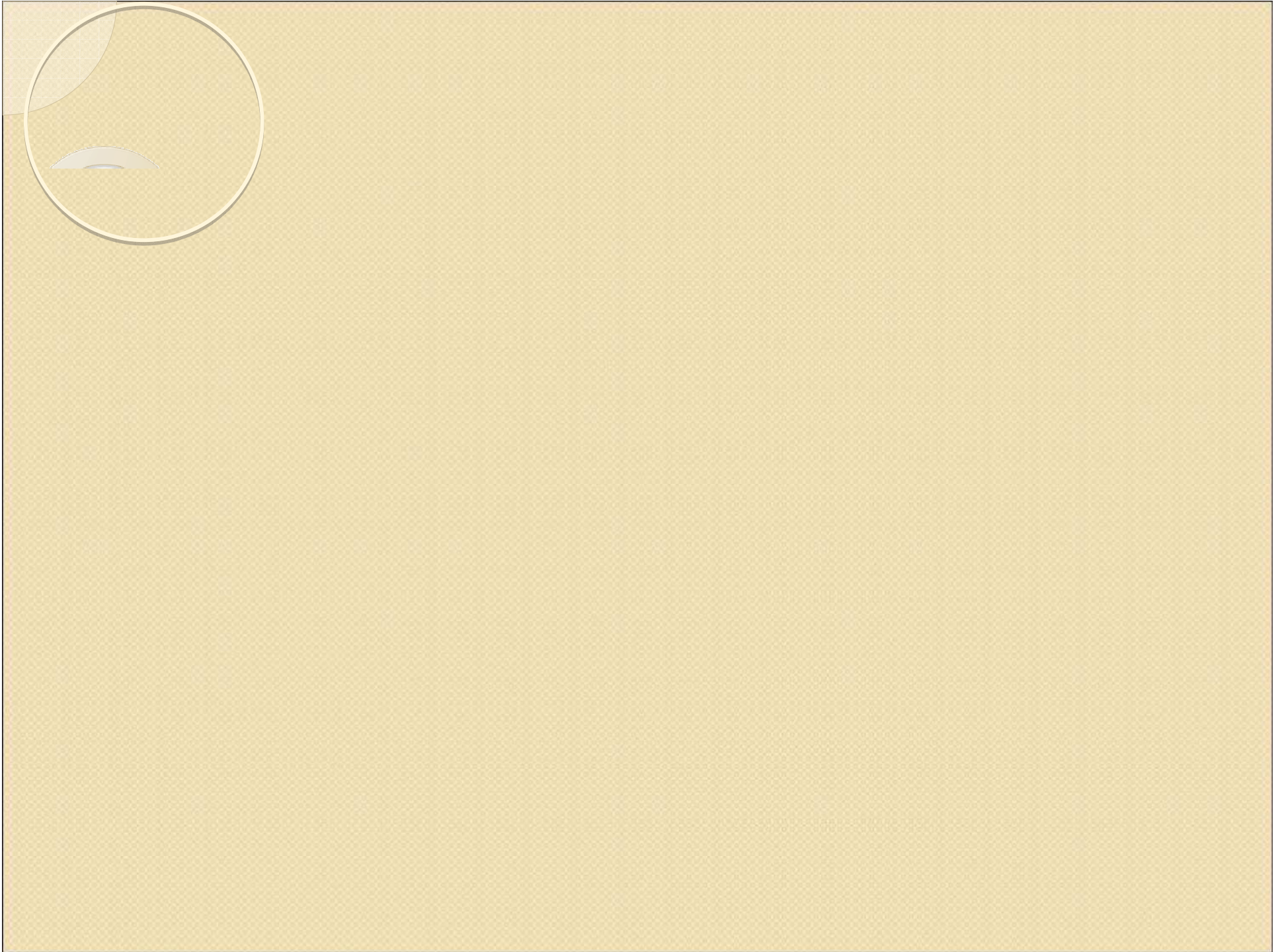




U – PANTHÉON - SORBONNE – 1  
UNIVERSITÉ PARIS 1

# Framework for Agile Methods Classification

Adrian Iacovelli & Carine  
Souveyet







# Problematic

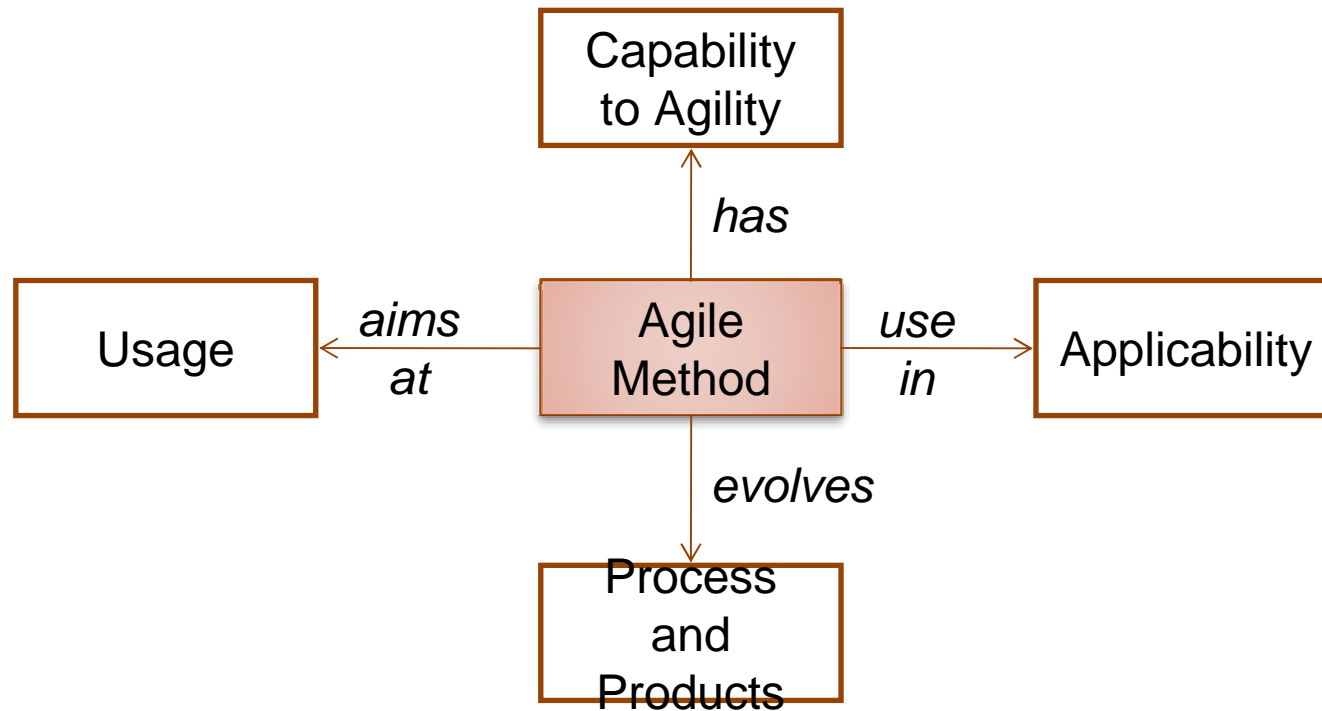
- What's the agile concept, how to capture it?
- How to integrate this concept to other methods?

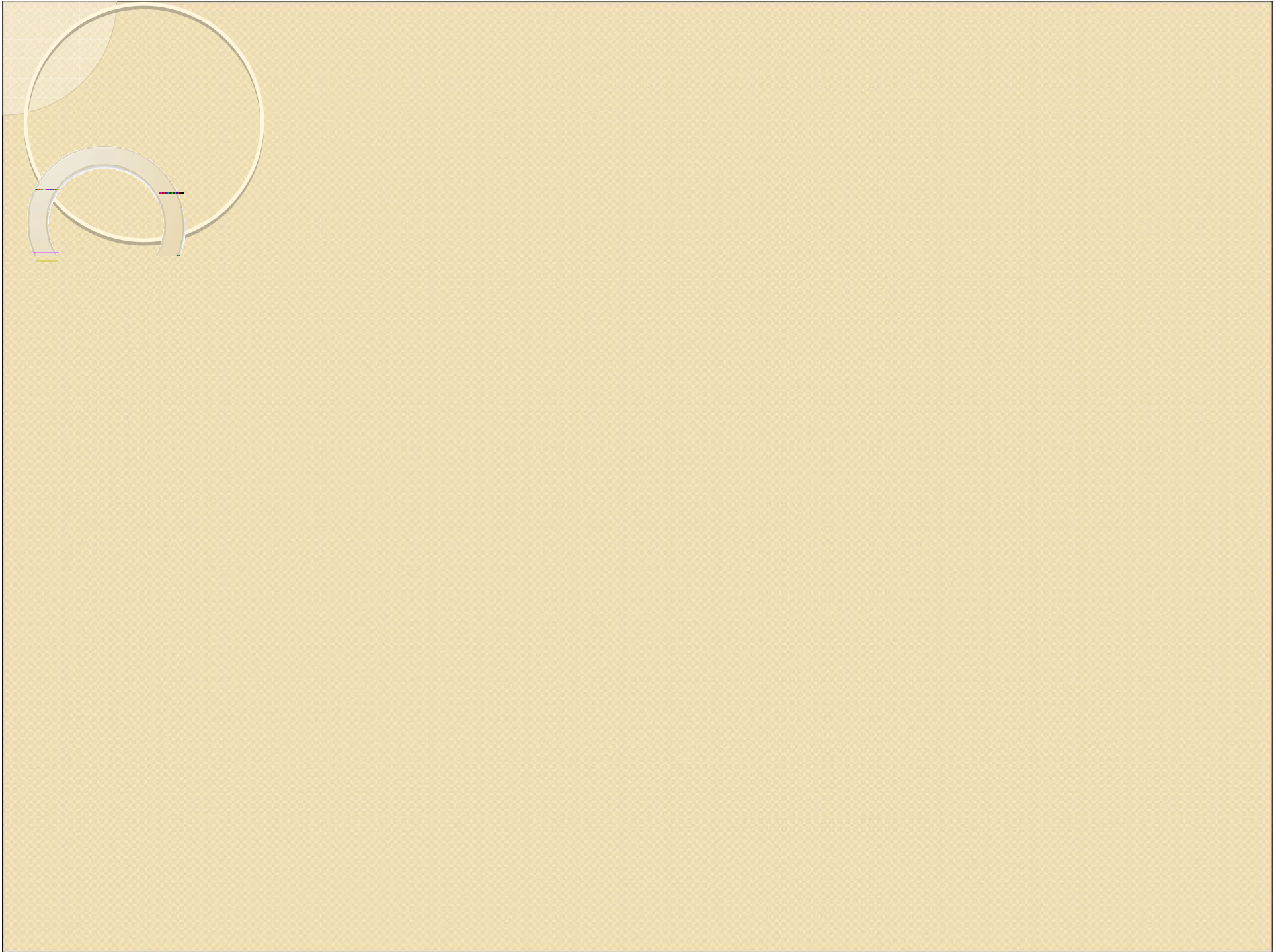


# Our Approach

- State of the art of agile methods:
  - Adaptive Software Development (ASD)
  - Agile Modeling (AM)
  - Crystal
  - Dynamic System Development Method (DSDM)
  - eXtrem Programming (XP)
  - Feature Driven Development (FDD)
  - Pragmatic Programming(PP)
  - Scrum
- Definition of a framework for comparing agile methods.
- Agile methods Classification
- Method fragment approach


# Framework





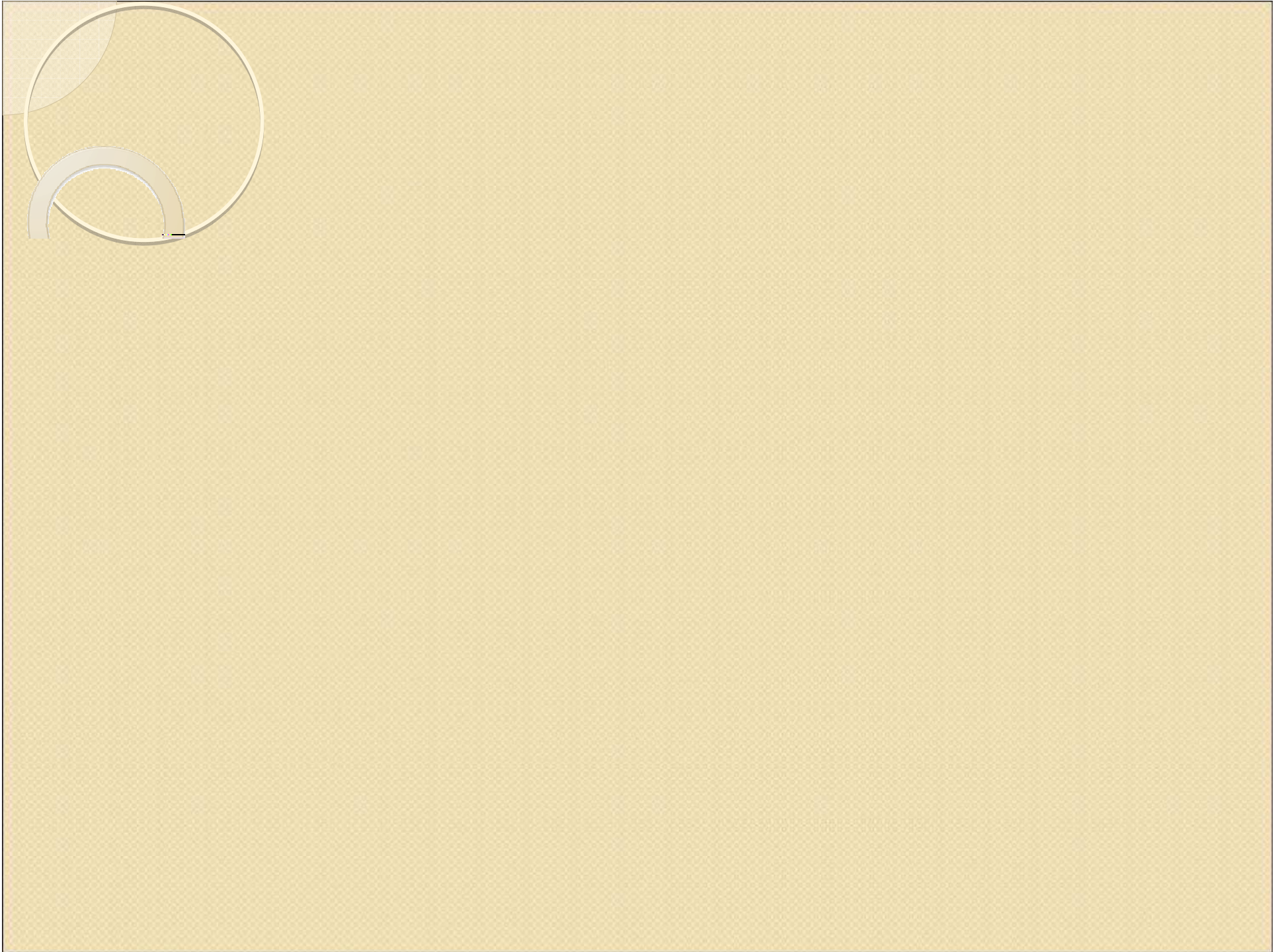
# Capability to Agility View

- **What** is the part of agility included in the method?

- 
- *Integration of the changes.*
  - *Collaborative.*
  - *Functional requirements can change.*
  - *Non functional requirement can change.*
  - *Testing politic.*
  - *Work plan can change.*
  - *Light weighted.*
  - *Change indicators.*
  - *Reactivity.*
  - *Short iterations.*
  - *Human resources can change.*
  - *People centered.*
  - *Refactoring politic.*
  - *Knowledge sharing.*

# Applicability View

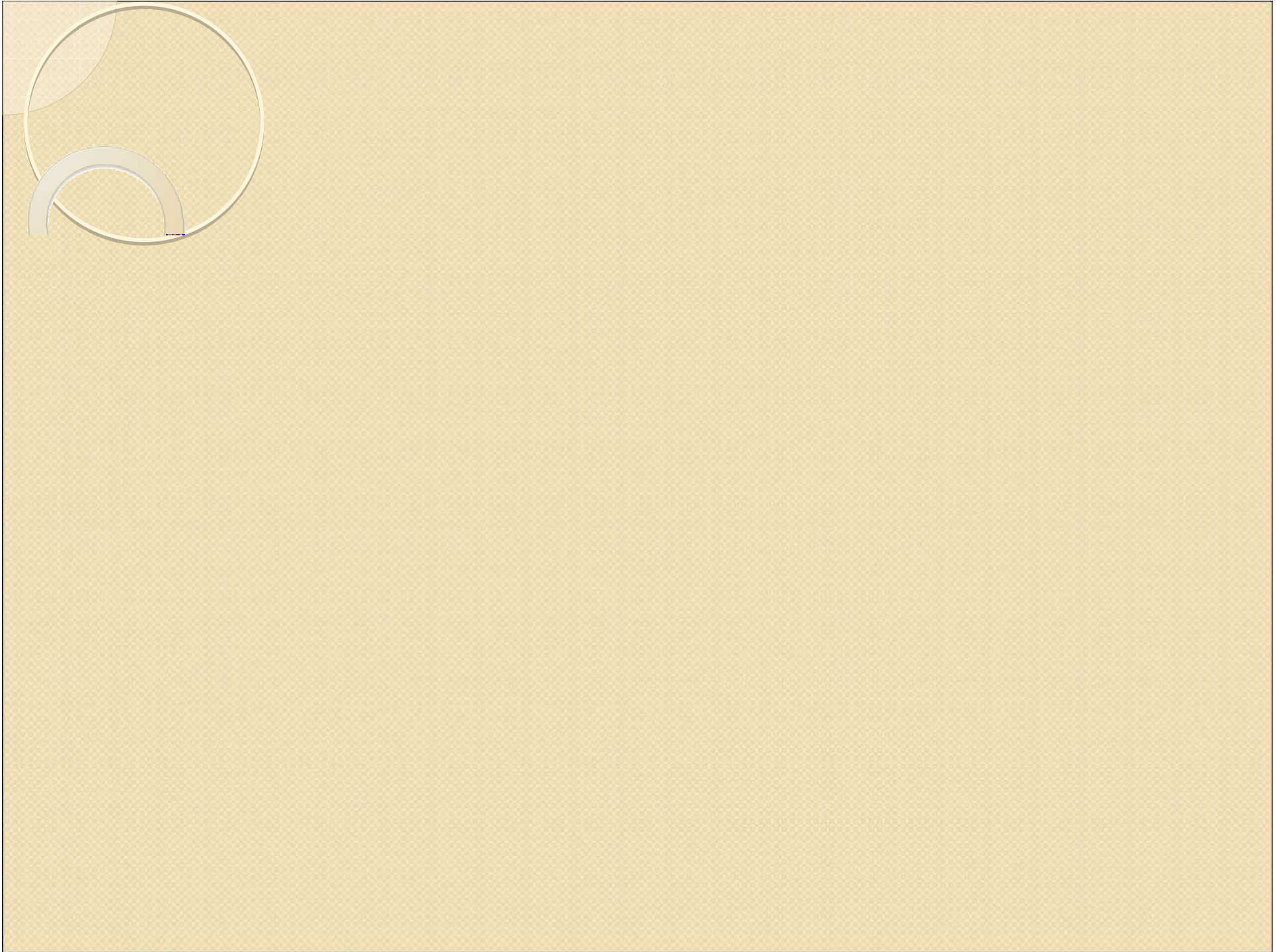
- **When** is the environment favorable to use the method?
- *Degree of interaction between the team members* → Open space
- *Degree of interaction with the customer* →
- *Degree of interaction with the end users.*
- *Degree of novelty integration.*
- *Project complexity.*
- *Project risks.*
- *Project size.*
- *Team organization.*
- *Team size.*







	ASD	AM	Crystal	DSDM	XP	FDD	PP	Scrum
<b>Usage view</b>								
Respect of delivering dates	True	False	True	True	False	True	True	True
Respect of the requirements	True	True	True	True	True	True	False	True
Respect of a quality level	True	False	False	False	False	False	True	False
End user satisfaction	False	False	False	True	False	False	False	False
Turbulent environment	False	True	False	True	True	False	True	True
Favourable to off shoring	True	False	True	True	False	False	False	True
Productivity gain	False	True	False	False	True	False	True	False
<b>Capability to agility view</b>								
Short iterations	False	True	False	False	True	True	True	True
Collaborative	True	True	True	True	True	False	True	False
People centred	True	True	True	True	True	False	False	False
Refactoring politic	False	True	False	True	True	False	True	False
Testing politic	True	False	False	True	True	False	True	True
Integration of the changes	True	True	False	True	True	False	True	True
Light weighted	False	True	False	False	True	True	True	True
FR can change	True	True	False	True	True	False	True	True
NFR can change	True	False	False	False	False	False	False	False
Work plan can change	False	True	False	False	True	False	True	False
Human resources can change	True	True	True	False	True	False	False	False
Change indicators	False	False	False	False	True	False	False	False
Re activity	Milestone	Iteration	Milestone	Iteration	Iteration	Project beginning	Iteration	Mile stone
Knowledge sharing	High	High	High	Low	High	Low	Low	Low
<b>Applicability view</b>								
Project size	Large	Small	Large	Large	Small	Large	Small	Large
Project complexity	High	Low	High	High	Low	High	Low	High
Project risks	High	Low	High	High	Low	High	Low	High
Team size	Large	Small	Small	Large	Small	Large	Small	Small
Interactions with customers	High	High	Low	High	High	Low	Low	Low
Interaction with end users	Low	Low	Low	High	Low	Low	Low	Low
Team members interactions	Low	High	High	High	High	Low	High	Low
Degree of novelty integration	High	Low	Low	High	High	Low	Low	Low
Team organization	Hierarchical	Self	Self	Hierarchical	Self	Hierarchical	Hierarchical	Self
<b>Process and products view</b>								
Activities covered by the method	rd, m, c, ut, it, st, at, qc	rd, m, c	m, cut, it, st	l, rd, m, c, ut, it, st, at, su	rd, m, c, ut, it, st	rd, m, c, ut, it, st, qc	rd, m, c, ut, it, st	rd, m, c, ut, it, st
Abstraction level of the guidelines	pm, pd	crg	pm, pd	pm, pd	pd, crg	pm, pd, crg	crg	pm
Products of the method activities	csc, exe, ut, it, st,	dm, csc, exe	csc, exe, ut, it, st	dm, csc, exe, ut, it,	csc, exe, ut, it, st	dm, csc, exe, ut, it,	dm, csc, exe, ut, it,	dm, csc, exe, ut, it,





# Method Fragment Approach

	<u>What</u>	<u>How</u>	<u>Why</u>	<u>When</u>
<b>Launching of the Project</b>		Feasibility and business study	Respect of delivering dates	Large and complex projects
<b>Management of the End Users</b>	People centered and reactivity	Involving users in the process activities	Increase end users satisfaction	Projects with a high interaction rate
<b>Quality</b>	Changes in NFR	Quality control activities and	Respect of a quality level	Complex and risky projects
<b>Tests</b>	Test politic	Testing activities and products	Productivity gain	All projects
<b>Refactoring</b>	Refactoring politic	Constant code review	Productivity gain	Project with low risks and complexity
<b>Knowledge Management</b>	People centered and knowledge sharing	Self organizing teams and high interaction rate	Productivity gain	Small teams with high interaction rate
<b>Agile Life Cycle</b>	Iterative and incremental life cycle	Short iterations and meetings with the customer	Fit turbulent environments	High interaction rate
<b>Change Indicator</b>	Manage the changes	Project velocity metric	Respect of delivering dates	All projects

# Fragment example

## **Change Indicator: Project Velocity**

Objectives. Manage changes and delivering dates by doing better estimations on development time.

Application Domain: Projects with a iterative and incremental life cycle.

Intention: Make more precise estimations  
<(estimations and real development time of functionalities of the previous iteration, estimations for the next iteration), calculate the ratio between the estimate time and the real time it takes for the previous iteration. Then apply this factor to the estimations of the next iterations.

- Project velocity = development time / estimate time of the previous iteration.
- Adjusted estimate development time of a functionality = estimate time X Project velocity.
- Adjust the work plan of the next iteration with the new estimations.



# Conclusion

- Method ingeniering: Method chunks.
- Build a method agile chunk data base.
- Bring agility to plan-driven methods.