澳 門 理 工 學 院

Instit nico de Macau

nnic Institute

Online Learning Materials

COMP223: Software Engineering

Agile Software Development (B)

Dr. Kim, Song-Kyoo (Amang)
Associate Professor,

Computer Science Program
MACAO POLYTECHNIC INSTITUTE
Macau, SAR

Session (Chapter 3) Objectives

- Agile methods
- Agile development techniques
- Agile project management
- Scaling agile methods

澳 門 理 工 學 院
Instituto Politécnico de Macau
Macao Polytechnic Institute

Scaling Agile Methods

Scaling Agile Methods (1/19)

- Agile methods have proved to be successful for small and medium sized projects that can be developed by a small co-located team.
- It is sometimes argued that the success of these methods comes because of improved communications which is possible when everyone is working together.
- Scaling up agile methods involves changing these to cope with larger, longer projects where there are multiple development teams, perhaps working in different locations.

Scaling Agile Methods (2/19)

- **Scaling out and scaling up**
  - ‘Scaling up’ is concerned with using agile methods for developing large software systems that cannot be developed by a small team.
  - ‘Scaling out’ is concerned with how agile methods can be introduced across a large organization with many years of software development experience.
  - When scaling agile methods it is important to maintain agile fundamentals:
  - Flexible planning, frequent system releases, continuous integration, test-driven development and good team communications.

Scaling Agile Methods (3/19)

- Practical problems with agile methods
  - The informality of agile development is incompatible with the legal approach to contract definition that is commonly used in large companies.
  - Agile methods are most appropriate for new software development rather than software maintenance.
  - Yet the majority of software costs in large companies come from maintaining their existing software systems.
  - Agile methods are designed for small co-located teams yet much software development now involves worldwide distributed teams.

Scaling Agile Methods (4/19)

- Contractual issues
  - Most software contracts for custom systems are based around a specification, which sets out what has to be implemented by the system developer for the system customer.
  - However, this precludes interleaving specification and development as is the norm in agile development.
  - A contract that pays for developer time rather than functionality is required.
  - However, this is seen as a high risk my many legal departments because what has to be delivered cannot be guaranteed.

Scaling Agile Methods (5/19)

- Agile methods and software maintenance
  - Most organizations spend more on maintaining existing software than they do on new software development.
  - So, if agile methods are to be successful, they have to support maintenance as well as original development.
  - Two key issues:
    - Are systems that are developed using an agile approach maintainable, given the emphasis in the development process of minimizing formal documentation?
  - Can agile methods be used effectively for evolving a system in response to customer change requests?
  - Problems may arise if original development team cannot be maintained.

Scaling Agile Methods (6/19)

- Agile maintenance
  - Key problems are:
    - Lack of product documentation
    - Keeping customers involved in the development process
    - Maintaining the continuity of the development team
  - Agile development relies on the development team knowing and understanding what has to be done.
  - For long-lifetime systems, this is a real problem as the original developers will not always work on the system.

Scaling Agile Methods (7/19)

- Agile and plan-driven methods:
  - Most projects include elements of plan-driven and agile processes. Deciding on the balance depends on:
    - Is it important to have a very detailed specification and design before moving to implementation?
    - If so, you probably need to use a plan-driven approach.
    - Is an incremental delivery strategy, where you deliver the software to customers and get rapid feedback from them, realistic?
    - If so, consider using agile methods.
    - How large is the system that is being developed?
    - Agile methods are most effective when the system can be developed with a small co-located team who can communicate informally.
    - This may not be possible for large systems that require larger development teams so a plan-driven approach may have to be used.

Scaling Agile Methods (8/19)

- Agile principles and organizational practice

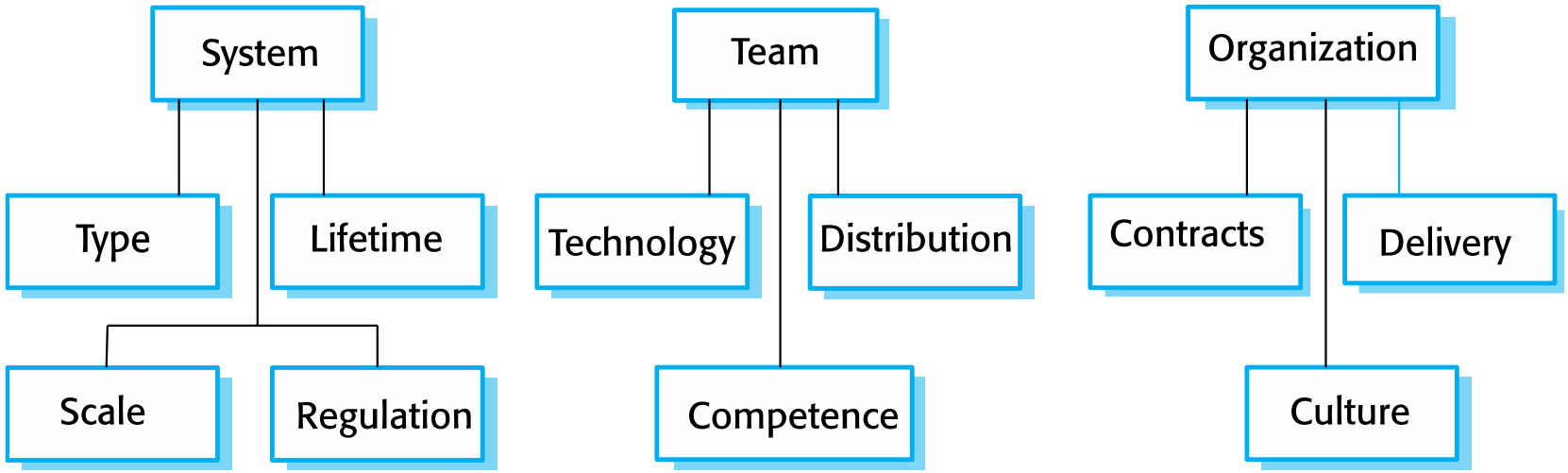
Principle	Practice
Customer involvement	This depends on having a customer who is willing and able to spend time with the development team and who can represent all system stakeholders. Often, customer representatives have other demands on their time and cannot play a full part in the software development. Where there are external stakeholders, such as regulators, it is difficult to represent their views to the agile team.
Embrace change	Prioritizing changes can be extremely difficult, especially in systems for which there are many stakeholders. Typically, each stakeholder gives different priorities to different changes.
Incremental delivery	Rapid iterations and short-term planning for development does not always fit in with the longer-term planning cycles of business planning and marketing. Marketing managers may need to know what product features several months in advance to prepare an effective marketing campaign.

Scaling Agile Methods (9/19)

- Agile principles and organizational practice

Principle	Practice
Maintain simplicity	Under pressure from delivery schedules, team members may not have time to carry out desirable system simplifications.
People not process	Individual team members may not have suitable personalities for the intense involvement that is typical of agile methods, and therefore may not interact well with other team members.

- Agile and plan-based factors



Scaling Agile Methods (10/19)

- **System issues**
  - How large is the system being developed?
    - Agile methods are most effective a relatively small co-located team who can communicate informally.
  - What type of system is being developed?
    - Systems that require a lot of analysis before implementation need a fairly detailed design to carry out this analysis.
  - What is the expected system lifetime?
    - Long-lifetime systems require documentation to communicate the intentions of the system developers to the support team.
  - Is the system subject to external regulation?
    - If a system is regulated you will probably be required to produce detailed documentation as part of the system safety case.

Scaling Agile Methods (11/19)

- **People and teams**
  - How good are the designers and programmers in the development team?
    - It is sometimes argued that agile methods require higher skill levels than plan-based approaches in which programmers simply translate a detailed design into code.
  - How is the development team organized?
    - Design documents may be required if the team is distributed.
  - What support technologies are available?
    - IDE support for visualization and program analysis is essential if design documentation is not available.

Scaling Agile Methods (12/19)

- **Organizational issues**
  - Traditional engineering organizations have a culture of plan-based development, as this is the norm in engineering.
  - Is it standard organizational practice to develop a detailed system specification?
  - Will customer representatives be available to provide feedback of system increments?
  - Can informal agile development fit into the organizational culture of detailed documentation?

Scaling Agile Methods (13/19)

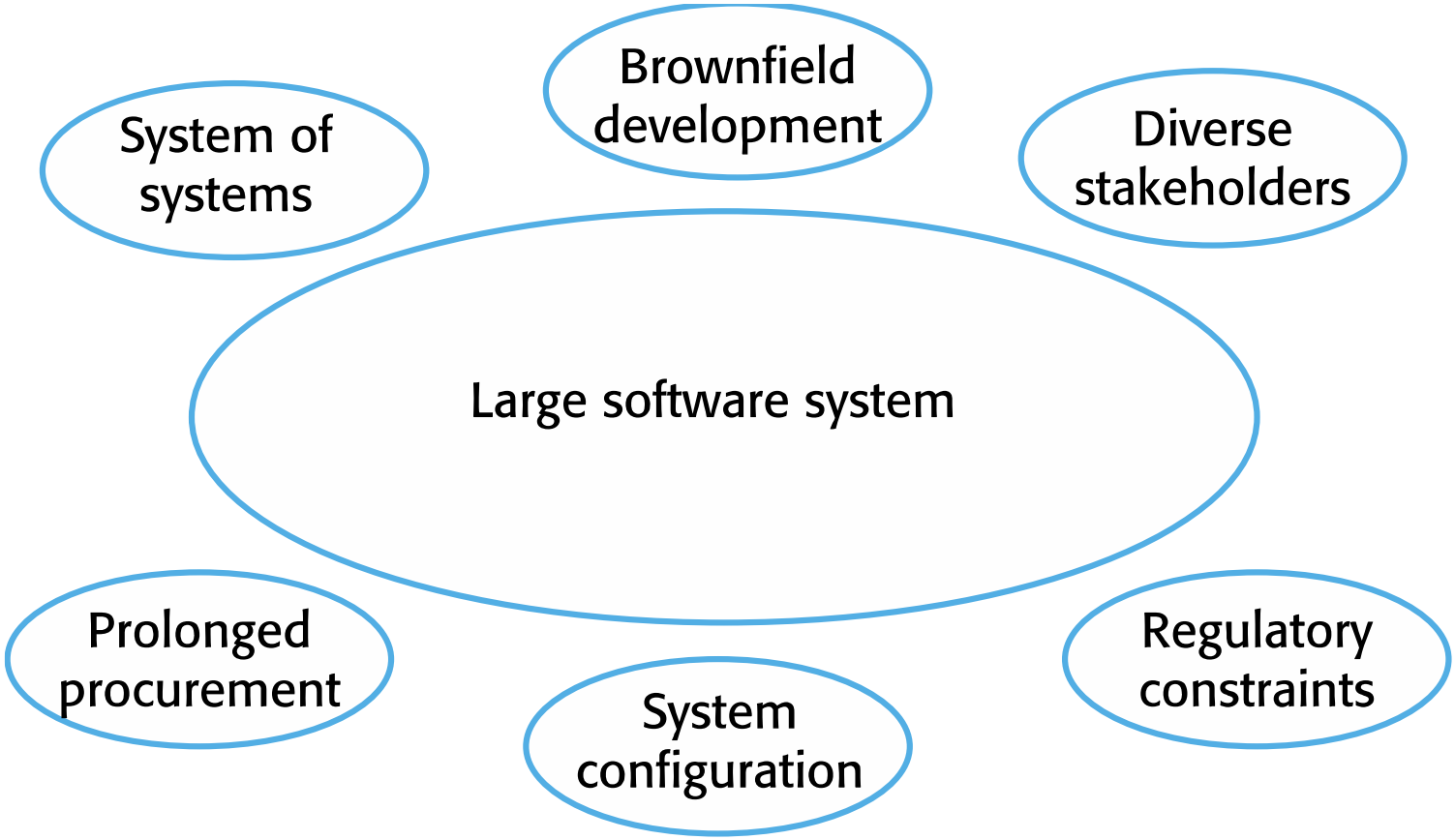
- Agile methods for large systems
  - Large systems are usually collections of separate, communicating systems, where separate teams develop each system.
  - Frequently, these teams are working in different places, sometimes in different time zones.
  - Large systems are ‘brownfield systems’, that is they include and interact with a number of existing systems.
  - Many of the system requirements are concerned with this interaction and so don’t really lend themselves to flexibility and incremental development.
  - Where several systems are integrated to create a system, a significant fraction of the development is concerned with system configuration rather than original code development.

Scaling Agile Methods (14/19)

- **Large systems** and their development processes are often constrained by external rules and regulations limiting the way that they can be developed.
- Large systems have a long procurement and development time.
  - It is difficult to maintain coherent teams who know about the system over that period as, inevitably, people move on to other jobs and projects.
- Large systems usually have a diverse set of stakeholders.
  - It is practically impossible to involve all of these different stakeholders in the development process.

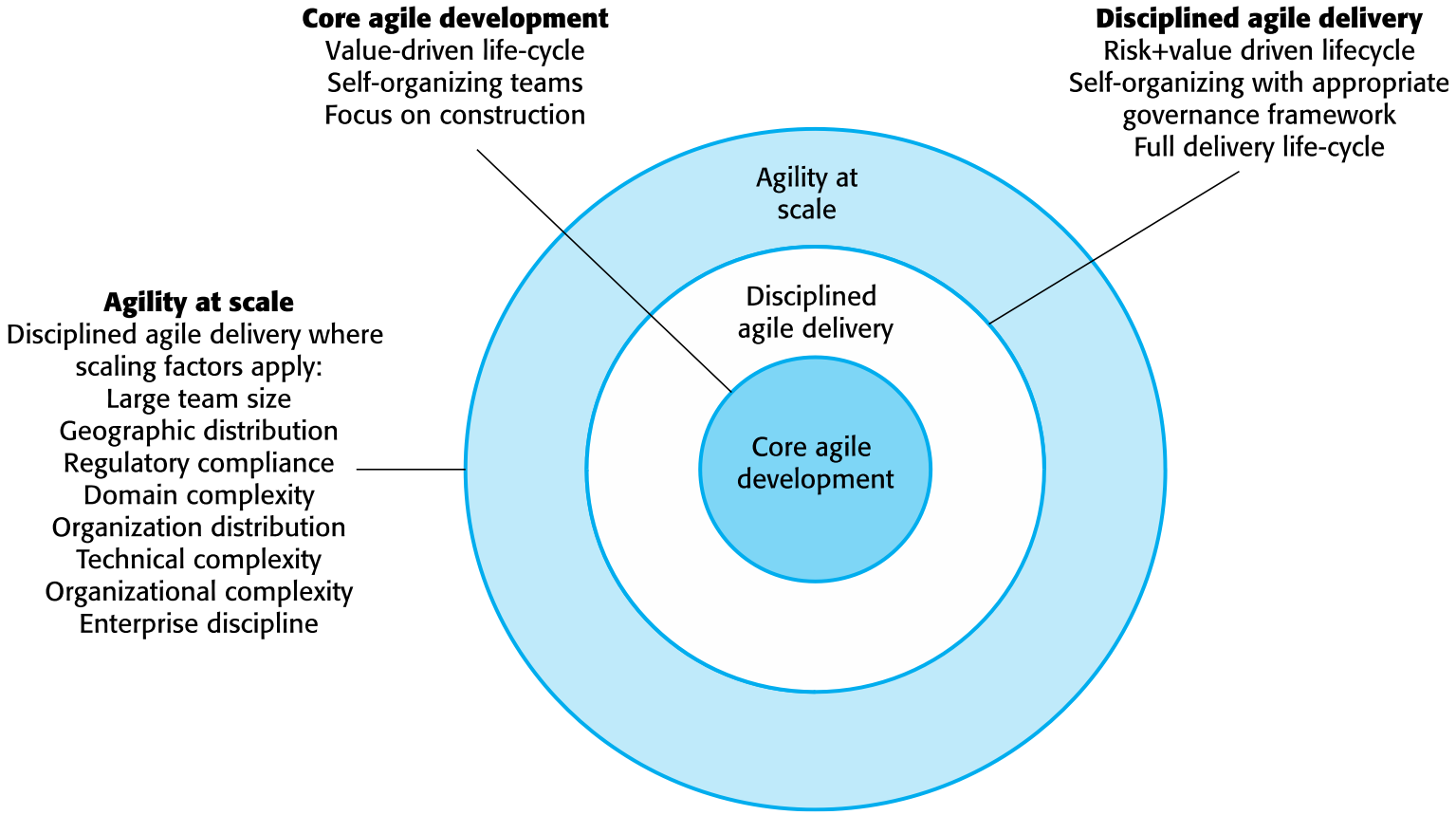
Scaling Agile Methods (15/19)

- Factors in large systems



Scaling Agile Methods (16/19)

- IBM's agility at scale model



Scaling Agile Methods (17/18)

- Scaling up to large systems
  - A completely incremental approach to requirements engineering is impossible.
  - There cannot be a single product owner or customer representative.
  - For large systems development, it is not possible to focus only on the code of the system.
  - Cross-team communication mechanisms have to be designed and used.
  - Continuous integration is practically impossible.
  - However, it is essential to maintain frequent system builds and regular releases of the system.

Multi-team Scrum (18/19)

- **Role replication:** Each team has a Product Owner for their work component and ScrumMaster.
- **Product architects:** Each team chooses a product architect and these architects collaborate to design and evolve the overall system architecture.
- **Release alignment:** The dates of product releases from each team are aligned so that a demonstrable and complete system is produced.
- **Scrum of Scrums:** There is a daily Scrum of Scrums where representatives from each team meet to discuss progress and plan work to be done.

Scaling Agile Methods (19/19)

- Agile methods across organizations
  - **Project managers** who do not have experience of agile methods may be reluctant to accept the risk of a new approach.
  - **Large organizations** often have quality procedures and standards that all projects are expected to follow and these are likely to be incompatible with agile methods.
  - Agile methods seem to work best when team members have a relatively high skill level.
  - However, within large organizations, there are likely to be a wide range of skills and abilities.
  - There may be **cultural resistance** to agile methods

Session Summary (1/2)

- **Agile methods** are incremental development methods that focus on rapid software development, frequent releases of the software, reducing process overheads by minimizing documentation and producing high-quality code.

- Agile development practices include
  - User stories for system specification
  - Frequent releases of the software,
  - Continuous software improvement
  - Test-first development
  - Customer participation in the development team.

Session Summary (2/2)

- Scrum is an agile method that provides a project management framework and it is centered round a set of sprints, which are fixed time periods when a system increment is developed.

- Many practical development methods are a mixture of plan-based and agile development.

- Scaling agile methods for large systems is difficult.
  - Large systems need up-front design and some documentation and organizational practice may conflict with the informality of agile approaches.


SCAN ME
SCAN ME