《软件工程》大程分组及其功能模块名单　　2024.3.11

组号	职责	姓名	学号	备注	组号	职责	姓名	学号	备注	模块名称
A1	组长	林方芊	3210100527		B1	组长	吴迪	3210105557		基础信息管理组
	组员	黄静彪	3200105271			组员	陈科睿	3210104320		
	组员	李杭奇	3210104821			组员	卢峰杰	3210102198		
	组员	刘志化	3230400064			组员	郑浩博	3210105321		
组号	职责	姓名	学号		组号	职责	姓名	学号		自动排课组
A2	组长	谢瑞航	3210106035		B2	组长	刘佳星	3210106007		
	组员	李心羽	3210104749			组员	李力扬	3210105647		
	组员	文博韬	3210102562			组员	王程业	3210101733		
	组员	项峥	3210102501			组员	俞心宇	3210104724		
	组员	胡炘炎	3210102517			组员	潘臻琦	3210102495		
组号	职责	姓名	学号		组号	职责	姓名	学号		智能选课组
A3	组长	薛杰怀	3210100662	A大组长	B3	组长	董冬	3210104573		
	组员	王一哲	3210102169			组员	张汉宸	3210106029		
	组员	张匡令	3210104612			组员	栗威	3210106175		
	组员	胡家齐	3210104424			组员	陈书陶	3210105352	B大组长	
	组员	陈艺真	3210300493			组员	郑维康	3210102381		
组号	职责	姓名	学号		组号	职责	姓名	学号		论坛交流组
A4	组长	黄琲	3210104881		B4	组长	唐朝	3210102187		
	组员	展翼飞	3190102196			组员	赵子炎	3210105581		
	组员	何永瑞	3230400061			组员	孟澍	3210101819		
	组员	赵元康	3210106046			组员	陈苇远	3210105677		
						组员	钱闻博	3210100736		

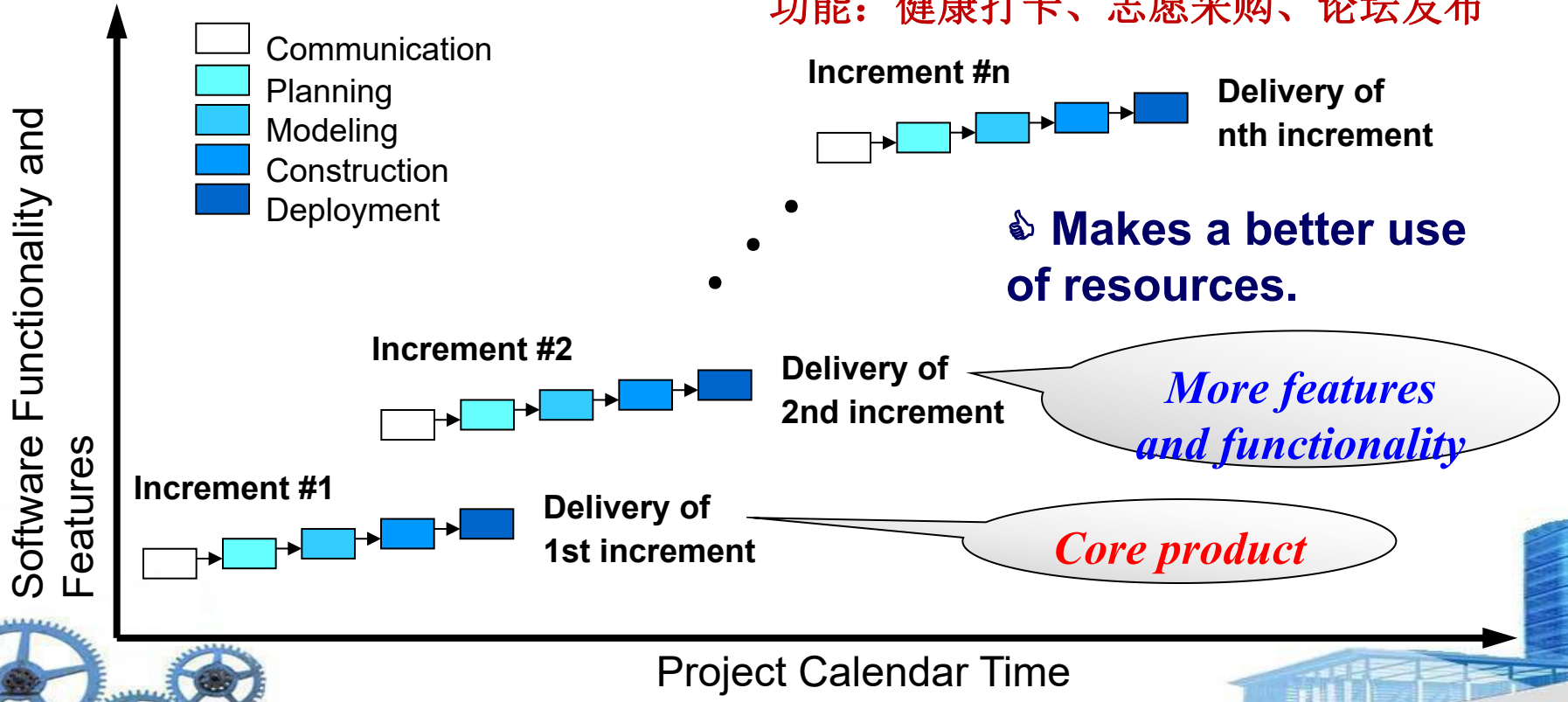
Ch.4 Process Models (Cont.)

March 11, 2024

4.1.2 Incremental (增量) Process Models

- **The Incremental Model**

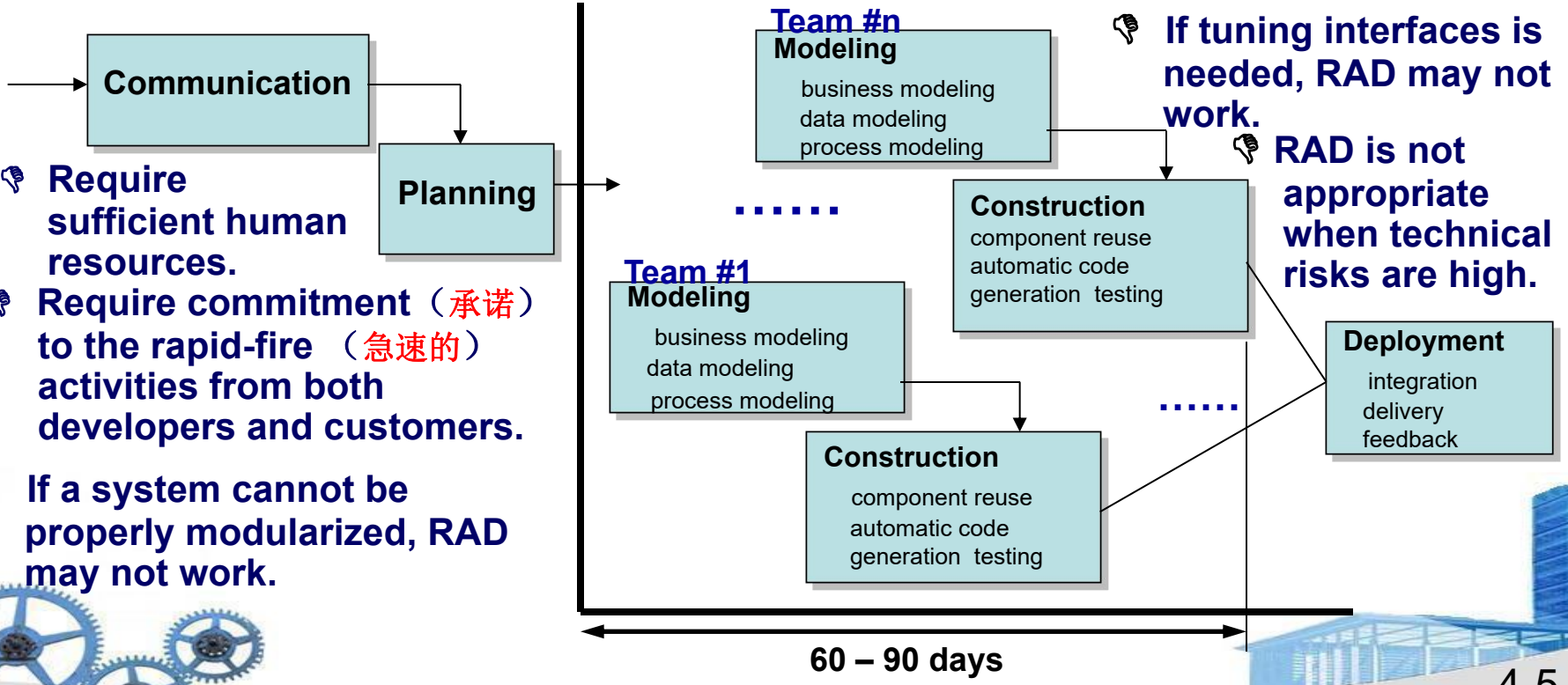
Ex. 四年前大程—**EMSS**（疫情监测与服务）
功能：健康打卡、志愿采购、论坛发布



Makes a better use of resources.

4.1.2 Incremental Process Models

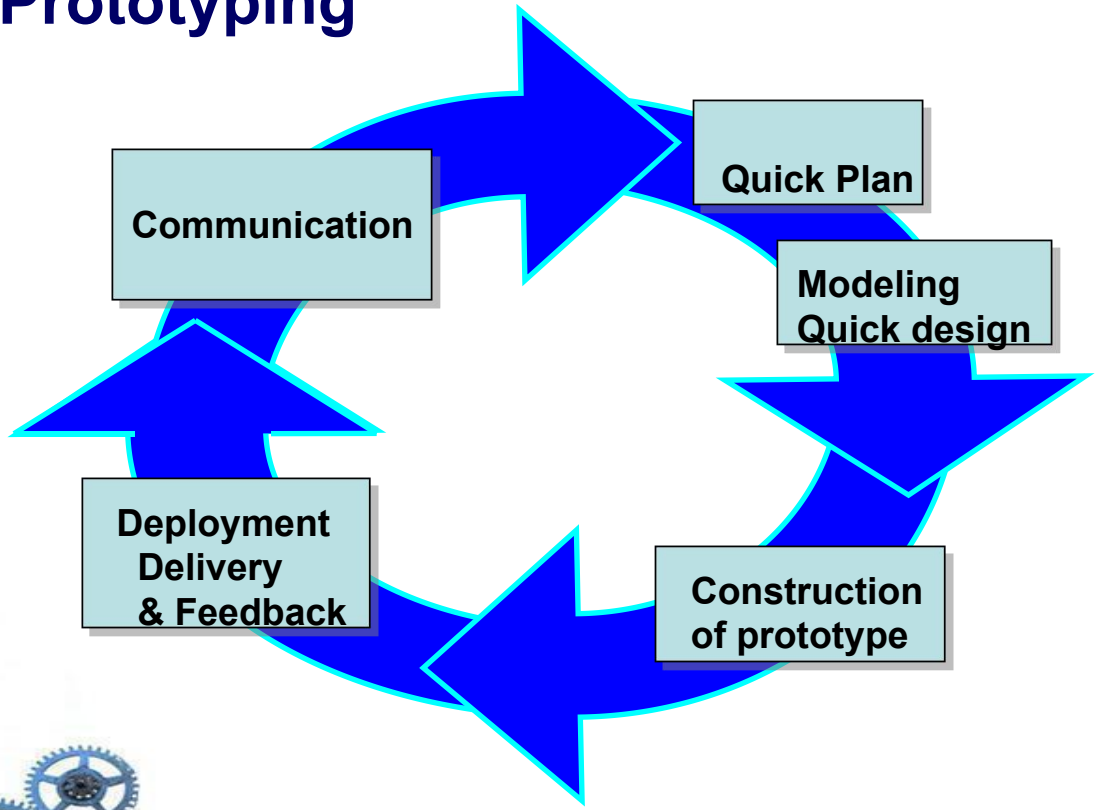
- The Rapid Application Development (RAD) Model



- Require sufficient human resources.
- Require commitment（承诺）to the rapid-fire （急速的）activities from both developers and customers.
- If a system cannot be properly modularized, RAD may not work.
- If tuning interfaces is needed, RAD may not work.
- RAD is not appropriate when technical risks are high.

4.1.3 Evolutionary Process Models

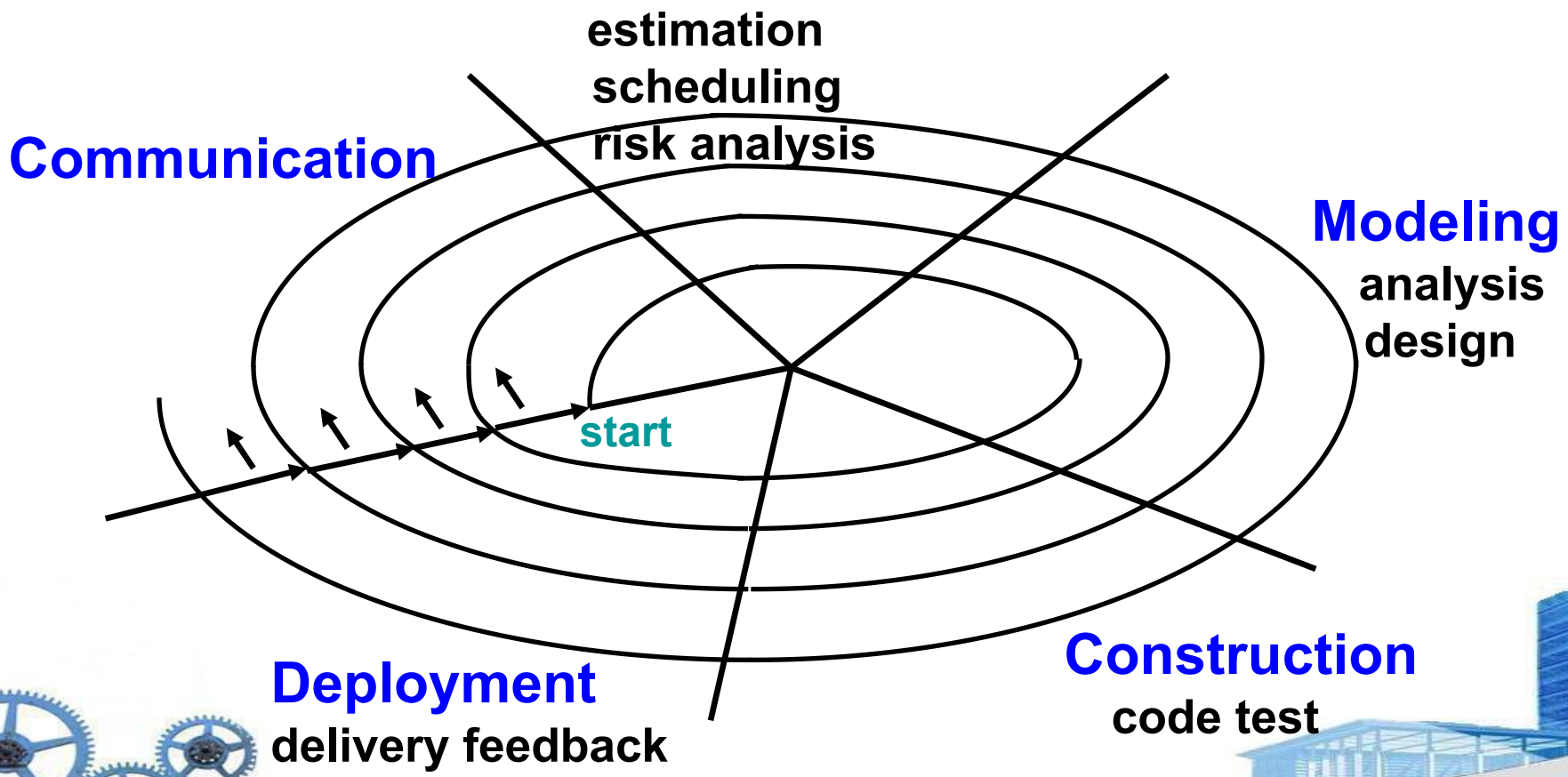
- **Prototyping**



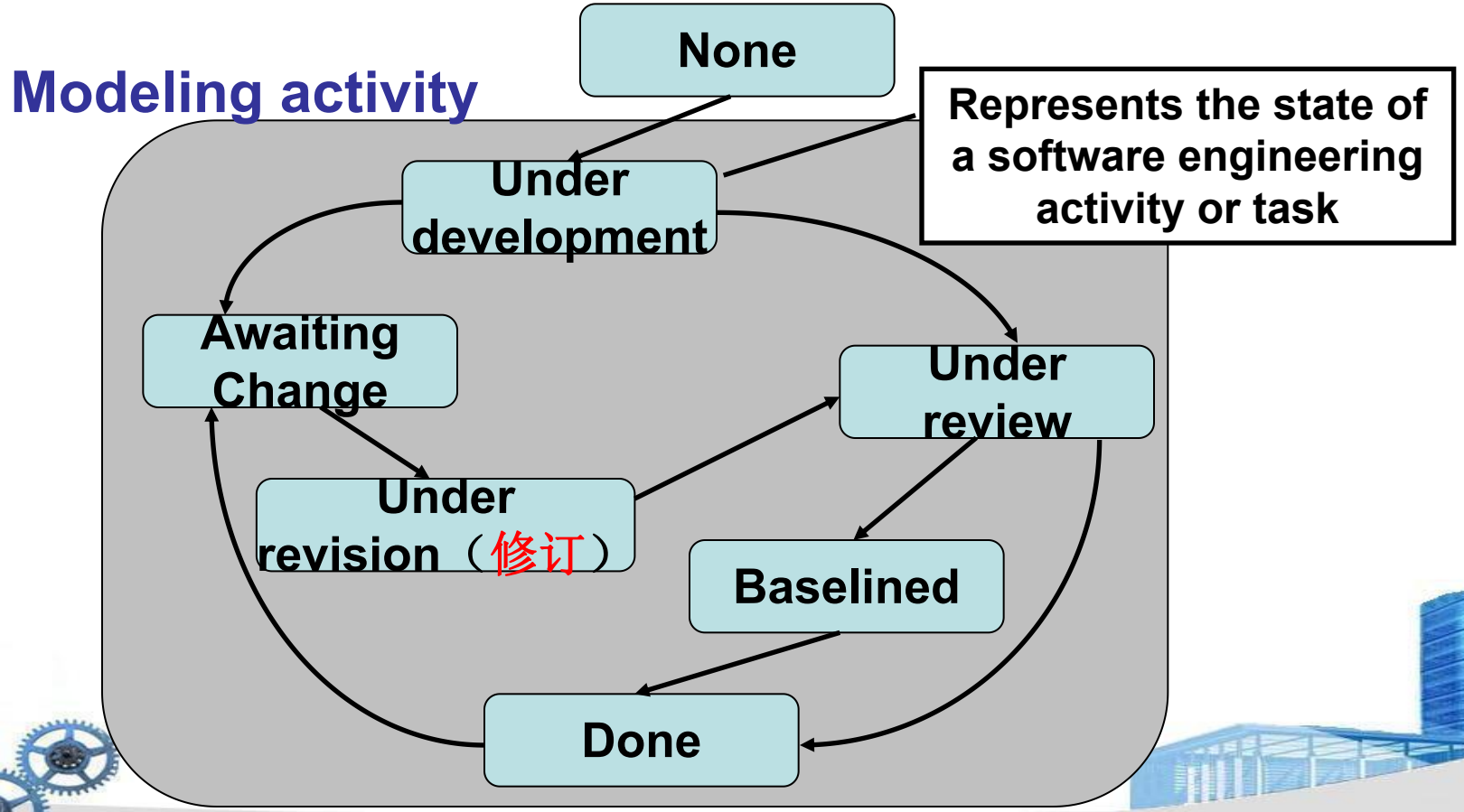
Ex. 林群书---整数智能

- Good first step when customer has a legitimate need, but is clueless about the details
- The prototype（原型）must be thrown away

Spiral Model



4.1.4 Evolutionary Process Models
Concurrent（协同）Development Model



4.1.4 Evolutionary Process Models

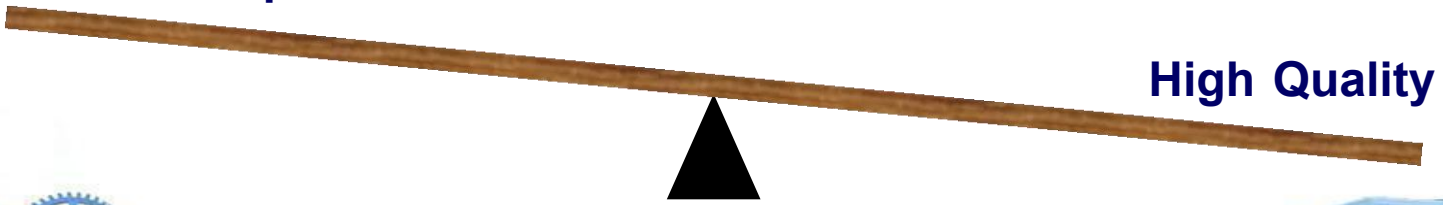
- **The Concurrent （协同） Development Model**

➢ Defines a series of events that will trigger transitions from state to state for each of the activities, actions or tasks.

➢ Especially good for client/server applications.

➢ Defines a network of activities instead of linear sequence of events.

Flexibility
Extensibility
Speed of development



4.2 Specialized（专用的） Process Models

- **Component based development** — the process to apply when reuse is a development objective

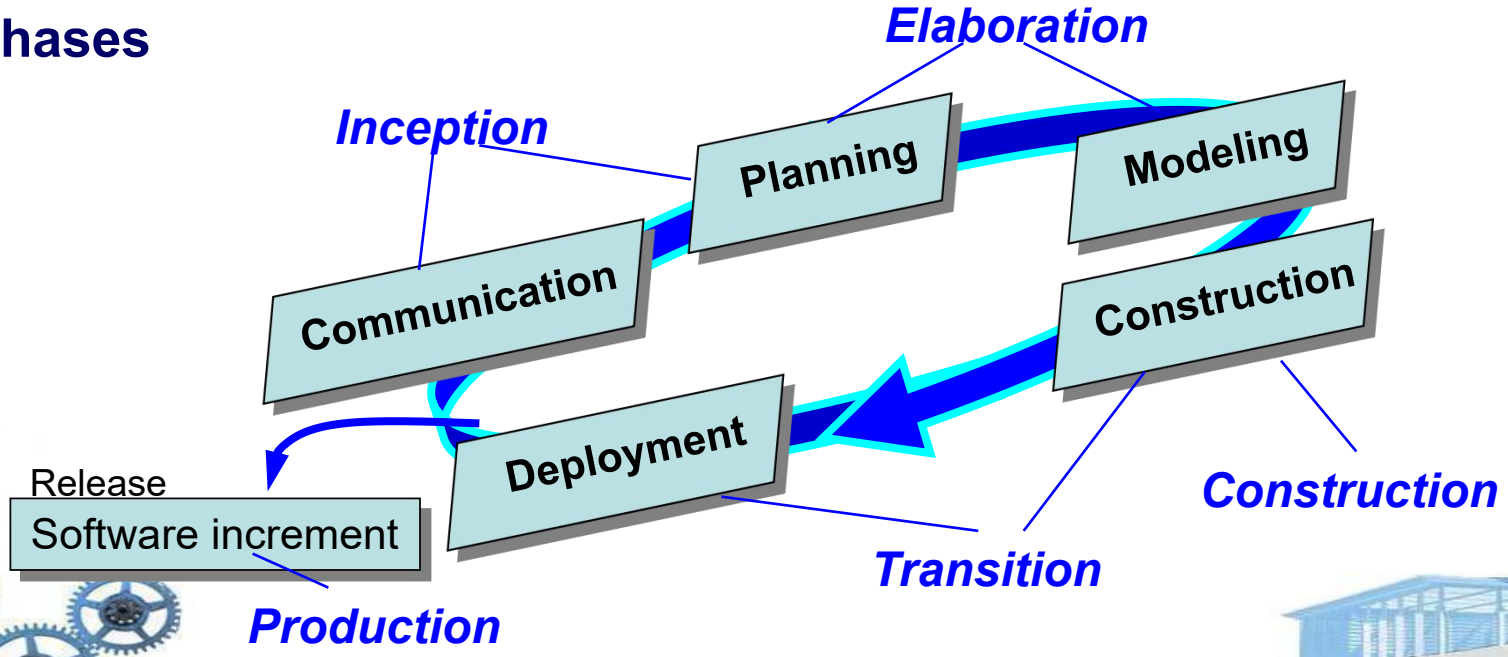
- **Formal（形式化） methods** — emphasizes the mathematical specification of requirements（See Ch.28）

- **Aspect-Oriented（面向方面） Software Development** — provides a process and methodological approach for defining, specifying, designing, and constructing aspects

e.g. Interface type

4.3 The Unified Process

- A "**use-case** driven, **architecture**-centric, **iterative** and **incremental**" software process closely aligned with the **U**nified **M**odeling **L**anguage (**UML**，统一建模语言)
- **Phases**



4.3 The Unified Process

• Work Products

Inception phase

- Vision document
- Initial use-case model
- Initial project glossary
- Initial business case
- Initial risk assessment

- Project plan
  phases and iterations

- Business model

- Prototypes

Elaboration phase

- Use-case model
- Functional and non-functional requirements
- Analysis model
- Software architecture description
- Executable architectural prototype
- Preliminary design model
- Revise risk list
- Project plan
  iteration plan, workflow, milestones
- Preliminary user manual

Construction phase

- Design model
- Software components
- Integrated software increment
- Test plan
- Test cases
- Support documentation
  user
  installation
  increment

Transition phase

- Delivered software increment

- Beta test reports

- User feedback

4.4 Personal and Team Process Models

- **Personal Software Process (PSP)) Ex. WPS(求伯君)**

- Recommends five framework activities:
  1. Planning
  2. High-level design
  3. High-level design review
  4. Development
  5. Postmortem (后验)
- Stresses the need for each software engineer to identify errors early and as important, to understand the types of errors

4.4 Personal and Team Process Models

- **Team Software Process (TSP)** Open AI Sora团队
- Each project is “launched（开始）” using a “script” that defines the tasks to be accomplished

➢ Teams are self-directed

➢ Measurement is encouraged

➢ Measures are analyzed with the intent of improving the team process

Ch.31 Project Management Concepts

- **The 4P's**
  - *People* — the most important element of a successful project
  - *Product* — the software to be built
  - *Process* — the set of framework activities and software engineering tasks to get the job done
  - *Project* — all work required to make the product a reality

Stakeholders

- *Senior managers* who define the business issues that often have significant influence on the project. (如：大组长（PM）：A战队：薛杰怀，B战队：陈书陶）
- *Project (technical) managers* who must plan, motivate, organize, and control the practitioners who do software work(如：A4组长：黄琲，B2组长：吴迪).
- *Practitioners* who deliver the technical skills that are necessary to engineer a product or application(如：全体组员，如：A1林方芊、B3钱文博等).
- *Customers* who specify the requirements for the software to be engineered and other stakeholders who have a peripheral interest in the outcome (如：各高校教务处,政府教育局....）.
- *End-users* who interact with the software once it is released for production use (如：学生、老师、教务管理人员....）.

- **Software Teams**

How to lead?

How to organize?

How to collaborate?

How to motivate?

How to create good ideas?

What is the **MOI Model** for team leader?

- **Team Leader**

"Stay Hungry, Stay Foolish" by Steve Jobs

- *The MOI Model*
- ***Motivation.*** **The ability to encourage (by "push or pull") technical people to produce to their best ability**
- ***Organization.*** **The ability to mold existing processes (or invent new ones) that will enable the initial concept to be translated into a final product.**
- ***Ideas or innovation.*** **The ability to encourage people to create and feel creative even when they must work within bounds established for a particular software product or application.**

- **Software Teams**
  - ***The following factors must be considered when selecting a software project team structure ...***
    - the *difficulty of the problem* to be solved
    - the *size of the resultant program(s)* in lines of code or function points
    - the *time that the team will stay together* (team lifetime)
    - the *degree to which the problem can be modularized*
    - the *required quality and reliability* of the system to be built
    - the *rigidity of the delivery date*
    - the *degree of sociability (communication)* required for the project

• Organizational Paradigms→范式，范型

Demis Hassabis

- ***closed paradigm*** —structures a team along a traditional hierarchy of authority
- ***random paradigm***—structures a team loosely and depends on individual initiative of the team members
- ***open paradigm***—attempts to structure a team in a manner that achieves some of the controls associated with the closed paradigm but also ***much of the innovation*** that occurs when using the r***andom paradigm***
- ***synchronous (同步的)paradigm***—relies on the natural compartmentalization of a problem and organizes team members to work on pieces of the problem with little active communication among themselves

*suggested by **Constantine (康斯坦丁)** [Con93]*

破碎的

协调的

- **Avoid Team "Toxicity"**
  - A frenzied work atmosphere in which team members waste energy and lose focus on the objectives of the work to be performed.
  - High frustration caused by personal, business, or technological factors that cause friction among team members.
  - "Fragmented or poorly coordinated procedures" or a poorly defined or improperly chosen process model that becomes a roadblock to accomplishment.
  - Unclear definition of roles resulting in a lack of accountability and resultant **finger-pointing**.
  - "Continuous and repeated exposure to failure" that leads to a loss of confidence and a lowering of morale.

责任

士气，斗志

- **Agile Teams**
  - Team members must have trust in one another.
  - The distribution of skills must be appropriate to the problem.
  - Mavericks（害群之马） may have to be excluded from the team, if team cohesiveness is to be maintained.
  - Team is “self-organizing”
    - An adaptive team structure
    - Uses elements of Constantine（康斯坦丁）’s random, open, and synchronous paradigms
    - Significant autonomy

Team Coordination & Communication

- *Formal, impersonal approaches* include software engineering documents and work products (including source code), technical memos（备忘录）, project milestones（里程碑）, schedules, and project control tools (Ch. 23), change requests and related documentation, error tracking reports, and repository（贮藏处） data (see Ch.26).
- *Formal, interpersonal procedures* focus on quality assurance activities (Ch.25) applied to software engineering work products. These include status review meetings and design and code inspections.
- *Informal, interpersonal procedures* include group meetings for information dissemination（分发） and problem solving and “collocation of requirements and development staff.”
- *Electronic communication* encompasses electronic mail, electronic bulletin boards, and by extension, video-based conferencing systems.
- *Interpersonal networking* includes informal discussions with team members and those outside the project who may have experience or insight that can assist team members（ e.g. Github）.

The Product Scope

- Scope
  - *Context*. How does the software to be built fit into a larger system, product, or business context and what constraints are imposed as a result of the context?
  - *Information objectives*. What customer-visible data objects (Ch.8) are produced as output from the software? What data objects are required for input?
  - *Function and performance*. What function does the software perform to transform input data into output? Are any special performance characteristics to be addressed?
- Software project scope must be unambiguous and understandable at the management and technical levels.

樊麾二段，欧洲围棋冠军
2015.10以0：5负于AlphaGo

2016.3AlphaGo Lee以4：1胜李世石

2017.5.23~27， AlphaGo Zero
以3：0胜柯洁

2016.12.29~2017.1.4,Master以60:0胜中日韩顶级围棋手

- **Problem Decomposition**
  - Sometimes called *partitioning* or *problem elaboration*
  - Once scope is defined …
    - It is decomposed into constituent functions 组成的
    - It is decomposed into user-visible data objects

    or
    - It is decomposed into a set of problem classes
  - Decomposition process continues until all functions or problem classes have been defined

- **The Process**
  - Once a process framework has been established
    - Consider project characteristics
    - Determine the degree of rigor required
    - Define a task set for each software engineering activity
      - Task set =
        - Software engineering tasks
        - Work products
        - Quality assurance points
        - Milestones

• Melding the Problem and the Process

COMMON PROCESS FRAMEWORK ACTIVITIES	communication	planning	modeling	construction	deployment
Software Engineering Tasks					
Product Functions					
Text input					
Editing and formatting					
Automatic copy edit					
Page layout capability					
Automatic indexing and TOC					
File management					
Document production					

• The Project

- Projects get into trouble when …
  - Software people don't understand their customer's needs.
  - The product scope is poorly defined.
  - Changes are managed poorly.
  - The chosen technology changes.
  - Business needs change [or are ill-defined].
  - Deadlines are unrealistic.
  - Users are resistant.
  - Sponsorship is lost [or was never properly obtained].
  - The project team lacks people with appropriate skills.
  - Managers [and practitioners] avoid best practices and lessons learned.

• Common-Sense Approach to Projects

- *Start on the right foot.* This is accomplished by working hard (very hard) to understand the problem that is to be solved and then setting realistic objectives and expectations.
- *Maintain momentum.* The project manager must provide incentives(鼓励) to keep **turnover** of personnel to an absolute minimum, the team should emphasize quality in every task it performs, and senior management should do everything possible to stay out of the team's way.
- *Track progress.* For a software project, progress is tracked as work products (e.g., models, source code, sets of test cases) are produced and approved (using formal technical reviews) as part of a quality assurance activity.
- *Make smart decisions.* In essence, the decisions of the project manager and the software team should be to "keep it simple."
- *Conduct a postmortem（后验） analysis.* Establish a consistent mechanism for extracting lessons learned for each project.

- To Get to the Essence of a Project(W^5HH)
  - *W*hy is the system being developed?
  - *W*hat will be done?
  - *W*hen will it be accomplished?
  - *W*ho is responsible?
  - *W*here are they organizationally located?
  - *H*ow will the job be done technically and managerially?
  - *H*ow much of each resource (e.g., people, software, tools, database) will be needed?

Barry Boehm [Boe96]

- Critical Practices
  - Formal risk management
  - Empirical cost and schedule estimation
  - Metrics-based project management
  - Earned value tracking
  - Defect tracking against quality targets
  - People aware project management

Ch.5 Agile Development

Q1: What's the meaning of Agile?

The Manifesto（宣言） for Agile Software Development

---In 2001,Kent Beck and 16 other noted software developers, writers, and consultants state:

- "We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan

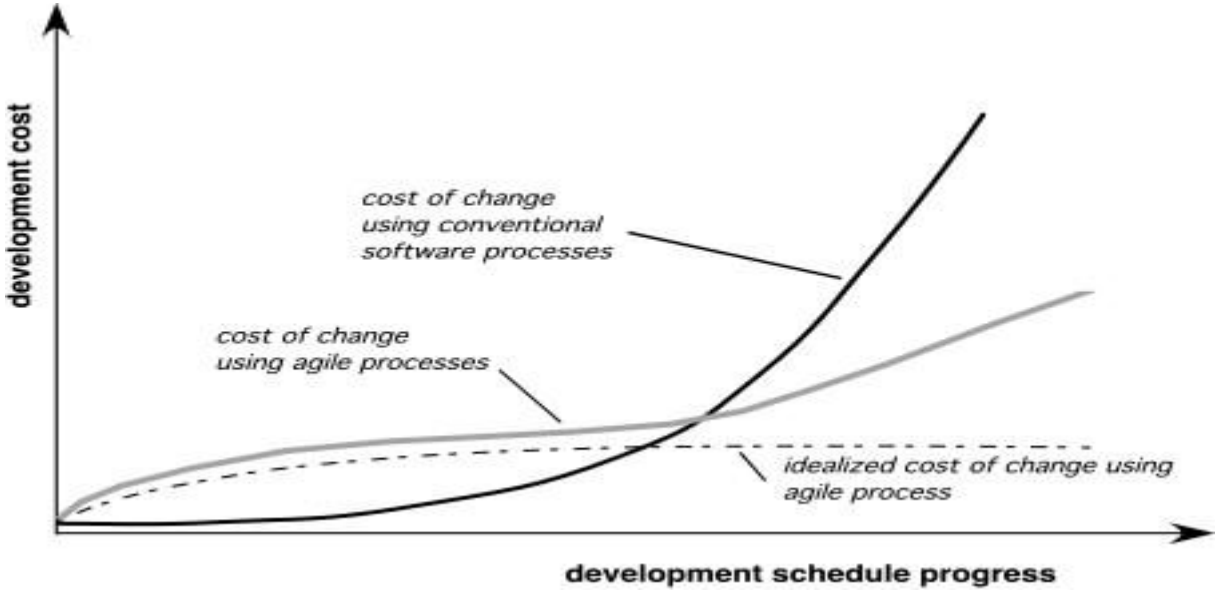
- That is, while there is value in the items on the right, we value the items on the left more."

- **What is “Agility”?**
  - Effective (rapid and adaptive) response to change
  - Effective communication among all stakeholders
  - Drawing the customer onto the team
  - Organizing a team so that it is in control of the work performed

  Yielding …

  - Rapid, incremental delivery of software

- **Agility and the Cost of Change**



- **An Agile Process**
  - Is driven by customer descriptions of what is required (scenarios)
  - Recognizes that plans are short-lived
  - Develops software iteratively with a heavy emphasis on construction activities
  - Delivers multiple ‘software increments’
  - Adapts as changes occur

Agility Principles - I 利用

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software(e.g. 选课功能).
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. **Business people** and **developers** must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face–to–face conversation.

Agility Principles - II

7. **Working software** is the **primary measure** of progress.
8. Agile processes promote **sustainable** (可持续的) development. The sponsors, developers, and users should be able to maintain a constant pace **indefinitely** (无限期地).
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity – the art of maximizing the amount of work not done – is essential.

（将没有做的部分最大化的艺术）

11. The best architectures, requirements, and designs emerge from self–organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Human Factors

- the process molds to the needs of the people and team, not the other way around
- key traits must exist among the people on an agile team and the team itself:
  - Competence.
  - Common focus.
  - Collaboration.
  - Decision-making ability.
  - Fuzzy problem-solving ability.
  - Mutual trust and respect.
  - Self-organization.

• Extreme Programming (XP)

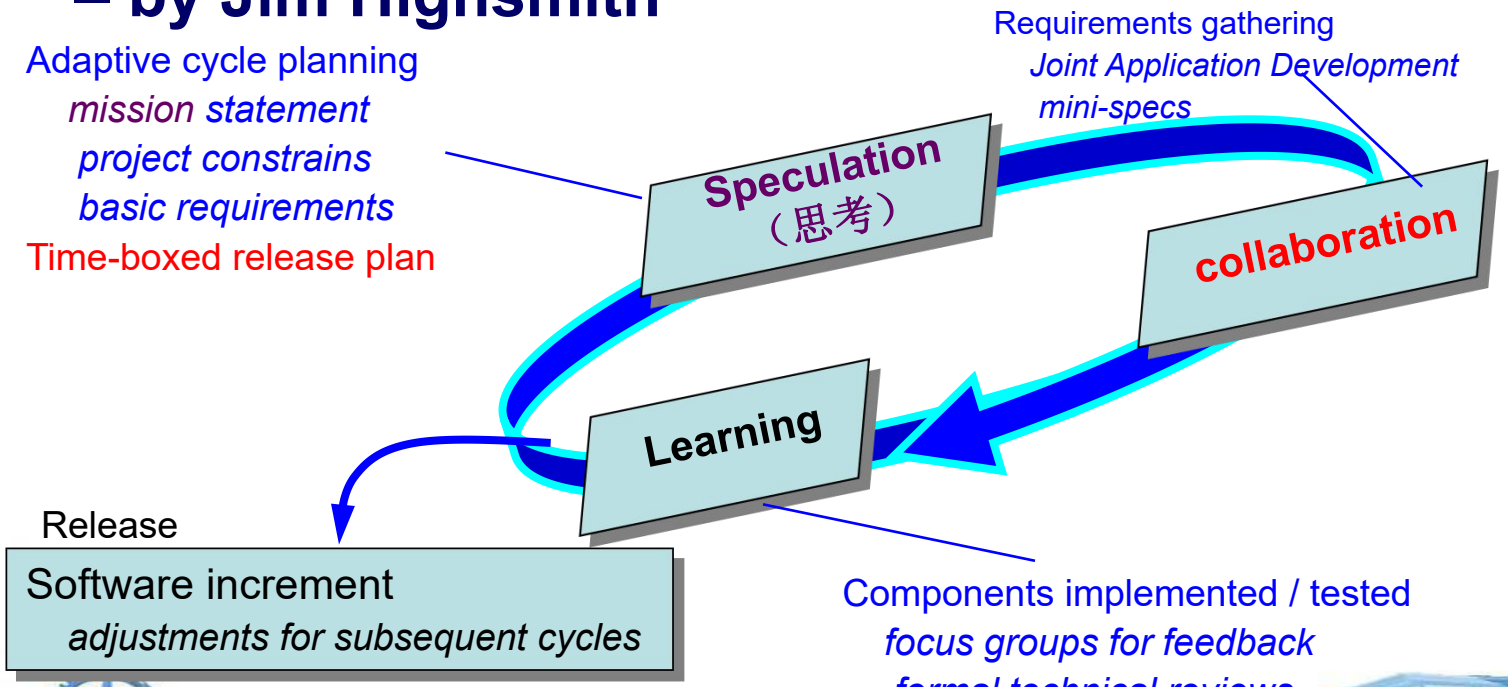
- The most widely used agile process, originally proposed by Kent Beck
- XP Planning
  - Begins with the creation of "***user stories***"(e.g.1）健康打卡；2）疫情服务；3）可视化；4）论坛，5）境外输入，etc.)
  - Agile team assesses each story and assigns a ***cost***
  - Stories are grouped to for a ***deliverable increment***
  - A ***commitment*** （*许诺*） is made on delivery date
  - After the first increment "***project velocity***" is used to help define subsequent delivery dates for other increments

Extreme Programming (XP)

- XP Design
  - Follows the ***KIS principle***
  - Encourage the use of ***CRC cards*** (see Chapter 8)
  - For difficult design problems, suggests the creation of "***spike solutions***"—a design prototype
  - Encourages "***refactoring***"—an iterative refinement of the internal program design
- XP Coding
  - Recommends the ***construction of a unit test*** for a store before coding commences
  - Encourages "***pair programming***"
- XP Testing
  - All ***unit tests are executed daily***
  - "***Acceptance tests***" are defined by the customer and executed to assess customer visible functionality

Extreme Programming (XP)

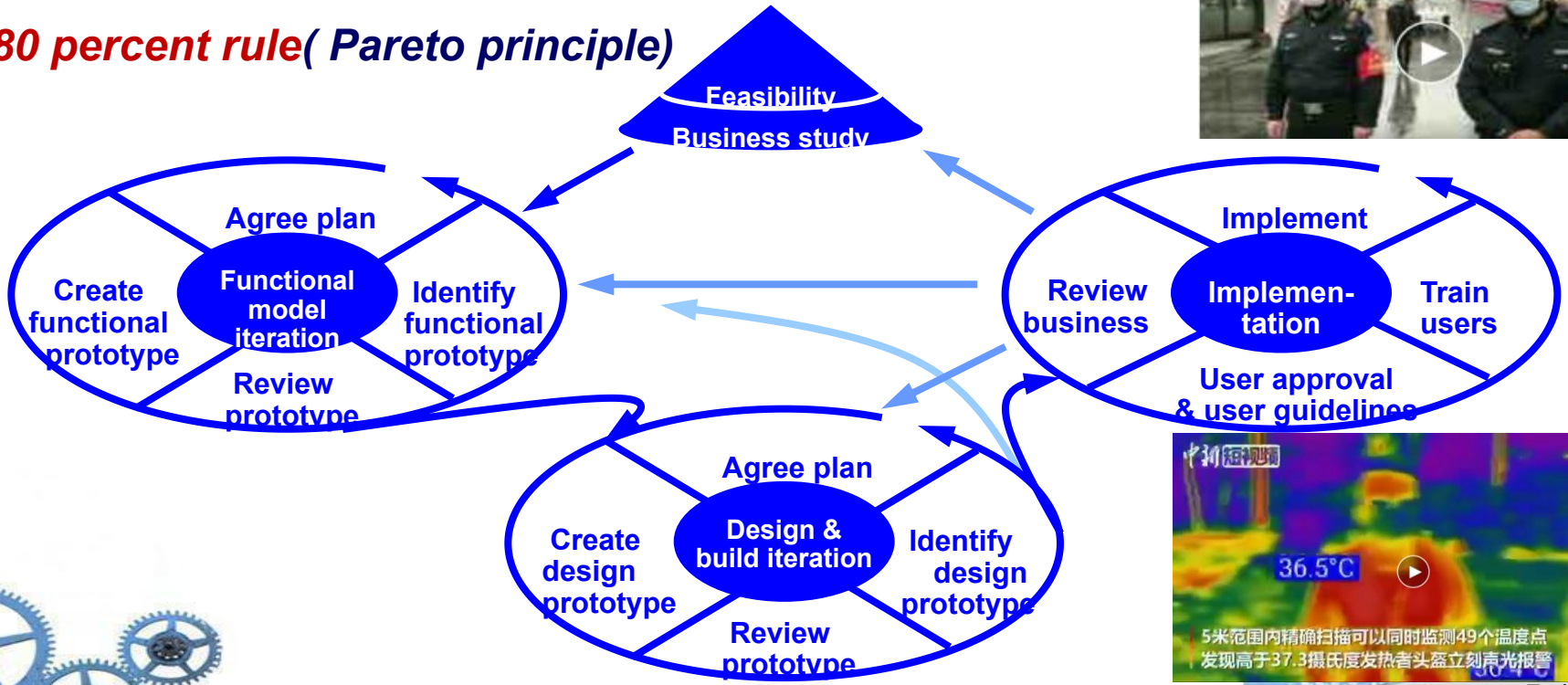
- **Adaptive Software Development (ASD)**
  – by Jim Highsmith

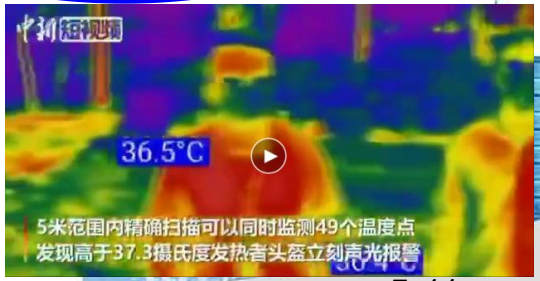


Agile Process Models

- Dynamic Systems Development Method (DSDM)
  - Promoted by DSDM Consortium (www.dsdm.org)

---80 percent rule(Pareto principle)





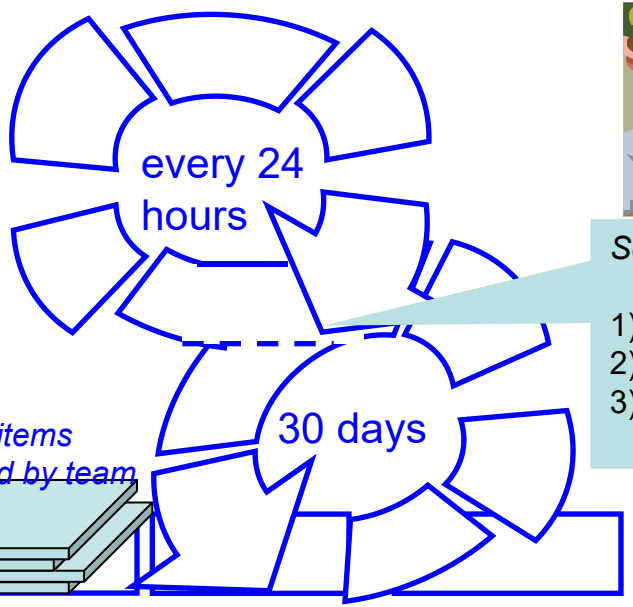
• Industrial XP (IXP)

- IXP has greater inclusion of management, expanded customer roles, and upgraded technical practices
- IXP incorporates six new practices:
  - Readiness assessment
  - Project community
  - Project chartering （承租）：团队自主对项目进行评估检查
  - Test driven management
  - Retrospectives (回顾)：增量交付后的技术评估
  - Continuous learning

- **Scrum**
  - Originally proposed by Schwaber and Beedle
  - Scrum—distinguishing features
    - Development work is partitioned into "***packets***" (打包)
    - ***Testing and documentation are on-going*** （不间断地） as the product is constructed
    - Work occurs in "***sprints***"（冲刺） and is derived from a "***backlog***" （待定项）of existing requirements
    - ***Meetings are very short*** and sometimes conducted **without chairs**
    - "***demos***" are delivered to the customer with the ***time-box*** *（时间段）* allocated

Agile Process Models

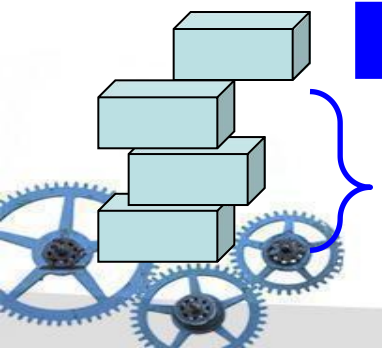
- **Scrum -- by Schwaber and Beedle**

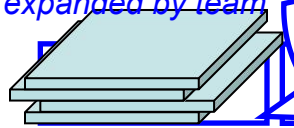


Scrum: 15-minute daily meeting. Team members respond to basics:
1) What did you do since last Scrum meeting?
2) Do you have any obstacles?
3) What will you do before next meeting?

冲刺　待定项

Sprint Backlog:
Feature(s) assigned to sprint Backlog items expanded by team





New functionality is demonstrated at end of sprint

Product Backlog:
Prioritized product features desired by the customer

Dynamic Systems Development Method

- Promoted by the DSDM Consortium (www.dsdm.org)
- DSDM—distinguishing features
  - Similar in most respects to XP
  - Nine guiding principles
    - Active user involvement is imperative （必要的）.
    - DSDM teams must be empowered（授权） to make decisions.
    - The focus is on frequent delivery of products.
    - Fitness for business purpose is the essential criterion for acceptance of deliverables.
    - Iterative and incremental development is necessary to converge（聚集） on an accurate business solution.
    - All changes during development are reversible（可逆的）.
    - Requirements are baselined at a high level
    - Testing is integrated throughout the life-cycle.

- **Agile Modeling**
  - Originally proposed by Scott Ambler
  - Suggests a set of agile modeling principles
    - Model with a purpose
    - Use multiple models
    - Travel light?

      （轻装上阵---学会舍弃）
    - Content is more important than representation
    - Know the models and the tools you use to create them
    - Adapt locally

- **Agile Unified Process**（AUP）
  - Each AUP iteration addresses these activities:
    - Modeling
    - Implementation
    - Testing
    - Deployment
    - Configuration and project management
    - Environment management

Task

- **Review** Ch.4-5, 31
- **Finish** “Problems and points to ponder” in Ch. 4-5, 31
- **Preview** Ch. 6,7,8
- Experimental Time on Tomorrow afternoon
  (March 12, Room 104)