| Model | Type | Main focus |
|---|---|---|
| Prototype Model | Software Development Model | Building a working model of system to understand Requirement better. |
| ~~Sprira~~ Spiral Model | Risk-Driven Development Model | Manage risks through repeated cycles |
| Extreme Programming (X P) | Agile Methodology | Improving code quality and quick adaptation to changes |
| Scrum | Agile Framework | Team collaboration and flexible project management ~~us~~using sprints |

| Process Style | Advantage | Disadvantages |
|---|---|---|
| Iterative (repead until user satisfied) | Easy to understand user need early | Can lead to frequent changes and delay in final system |
| Iterative + Risk analysis | Handles large, risk project well | Complex and expensive to manage. |
| Iterative, short development cycles | High customer satisfaction, better code quality | Requires highly skilled team and continuous user involvement |
| Iterative, time-boxed (2-4 week-sprints) | Very flexible, good for changing requirements | Can fail without strong teamwork and communicatio |

## Summary

Model   Advanced   Disadvantage focus

Protype – Frequent changes, poor documentation, and unrealistic user expectations.

Spiral – Complex, costly and requires expert risk ~~mang~~ management.

XP – High pressure, limited scalability, and poor documentation

Scrum – Role confusion, scaling difficulty and dependency on skilled teams.

## * Prototype Model Vs Sprial Model

| Point | Prototype Model | Sprial Model |
|---|---|---|
| 1. Focuse - | Focuses on building a prototype to understand user requirements | Focuses on risk management trough multiple spirals. |
| 2. Process Type | Simple iterative model with user feedback | Iterative and risk-driven model with structured phases |
| 3. Risk Handiling | Risk are not formally analyzed | Each phase includes detailed risk analysis and control |
| 4. Project size | Best for small or medium projects with unclear ~~requre~~ requirement | suitable for large and high risk projects. |
| 5. Cost & Complexity | Less costly and simple to manage | More costly and complex to implement |
| 6. Documentation | Less formal documentation | Requires extensive documentation and planning. |

2. Prototype

Point

1. Object

2. User Role

3. Iteration Length

4. team Involv

5. Docu

6. Suit

2. Prototype Model vs XP. (Extrem Programming)

| Point | Prototype Model | XP (Extrem Programming |
|---|---|---|
| 1. Object | To understand requirements through a working model | To improve code quality and adapt to changes quickly |
| 2. User Role | User gives feedback after seeing prototype | User stays continuously involved throughout development. |
| 3. Iteration Length | Iteration are long and flexible | ~~Iterat~~ Iterations (releases are short and fixed (1-3 weeks) |
| 4. team Involvement | Developers work on prototypes independently | Requires pair programmming and team collaboration |
| 5. Documentation | Prototype focuses less on documetation | XP. emphasizes code over documetation |
| 6. Suitable fo | Projects with unclear user needs | Projects needing frequent release and ~~cu~~customer feed back |

x Prototype Model Vs. Scrum

| Point | Prototype Model | Scrum |
|---|---|---|
| 1. Goal | Clarify requirements via prototype | Deliver working software in short sprints. |
| 2. Development cycle | Prototype refined until user satisfied | Development divided into fixed-length sprints (2-4 weeks) |
| 3. Team structure | No defined roles | Defined role Product Owner, Scrum Master, Teams |
| 4. Risk Handling | Risk not formally managed | Risk reduced via daily meetings and sprint reviews |
| 5. Documentation | Minimal ~~docu~~ documentation | Uses product backlog sprint backlog and burndown charts |
| 6. Flexibility | ~~Ed~~ Flexibility but unstructured | Highly structured Agile ~~form~~ frame~~work~~ |

4. Spiral

| Point |
|---|
| 1. Approach |
| 1. Nature |
| 2. Risk Am |
| 3. Team Si |
| 4. Custom Role |
| 5. Docum |
| 6. Deliver Time |

4. Spiral Model Vs XP

| Point | Spiral Model | XP |
|---|---|---|
| 1. Approach | Risk-based engineering approach | Agile and code-focused |
| 1. Nature | Risk driven and process-heavy | |
| 2. Risk Analysis | Formal risk analysis at every spiral | Minimal formal risk analysis. |
| 3. Team Size | Suitable for large teams project | Work best with small, skilled teams. |
| 4. Customer Role | Customer feedback ~~take~~ taken after each phase | Customer involved daily in the process |
| 5. Documentation | High level documentation | Very little documentation focus on working code |
| 6. Delivery Time | Longer development cya | |

5. Spiral Model vs Scrum

| Point | Spiral Model | Scrum |
|---|---|---|
| 1. Approach | Risk-based engineering approach | Iterative Agile famework using sprints |
| 2. Focus | Managing risk and costs | Managing teamwork and productivity |
| 3. Flexibility | Rigid with predefined phases ~~No specific team~~ | Highly flexible with changing priorities |
| 4. Roles | No specific team roles | Difined roles (Scrum Master, product Owner) |
| 5. Cost & time | Expensive and time-consuming | Cost-effective with shorten cycles |
| 6. Suitable for | Complex, high-risk projects | Adaptive fast moving software projects. |

6. XP vs

| Point |
|---|
| 1. Focus |
| 2. Iteration Name |
| 3. Technica Practices |
| 4. Documen |
| 5. Team |
| 6. Flexibi |

6. XP vs Scrum

| Point | XP. (Extrem Programming) | Scrum |
|---|---|---|
| 1. Focus | Focuses on coding practices and software quaility | Focuses on project management and teamwork. |
| 2. Iteration Name | Iteration called release cycle | Iteration called sprint. |
| 3. Technical Practices | Includes pair programming TDD refactoring, CI etc | Does not specify technical practices |
| 4. Documentation | Minimal documentation | Uses product and sprint backlogs. for tracking |
| 5. Team Size | Ideal for very small teams ($\leq 10$). | Can scale to larger Agile teams. |
| 6. Flexibility | Very flexible in coding and testing | Flexible in planning and project flow. |