# Software Engineering

## Software Engineering 02

# Software Process

### Software Process 1ක් යනු කුමක්ද?

- Software 1ක් නිපදවන සම්පත්, සීමාවන් (Constraints) හා ක්‍රියාකාරකම් අඩංගු වන, කිසියම් අනුපිළිවෙළකට අනුව සකස් කරන ලද කාර්යයන්(Tasks) වල එකතුවකි.
- එය Process 1 සෑදී තිබෙන ක්‍රියාකාරකම් පරීක්ෂා කිරීමට, අවබෝධ කර ගැනීමට, පාලනය කිරීමට හා වර්ධනය කර ගැනීමට ඉඩ ලබා දෙමින් අපගේ ක්‍රියාවන්ට මග පෙන්වයි.
- Software 1ක සියළුම අවධීන් අඩංගු වන හෙයින් සමහර අවස්ථාවලදී මෙම Process 1ට Lifecycle යන නම යෙදේ.

### Software Process Models:-

Process Model 1ක් නිර්මාණය කිරීම Process 1 තුළ ඇති නොගැලපීම්, අනවශ්‍ය දේ ආදිය හඳුනා ගැනීමට උපකාරී වේ. මෙවන් ගැටළු හඳුනා ගෙන නිවැරදි කර ගන්නා හෙයින් Process 1 වඩා කාර්යක්ෂම වේ. මෙවැනි Software Process Model කිහිපයක් නම්,

- Waterfall Model
- Prototyping Models
- Evolutionary Models
- The Spiral Model
- Formal Development
- Incremental Development
- Rapid Application Development
- Unified Process
- Agile Process
- Extreme Programming (XP)

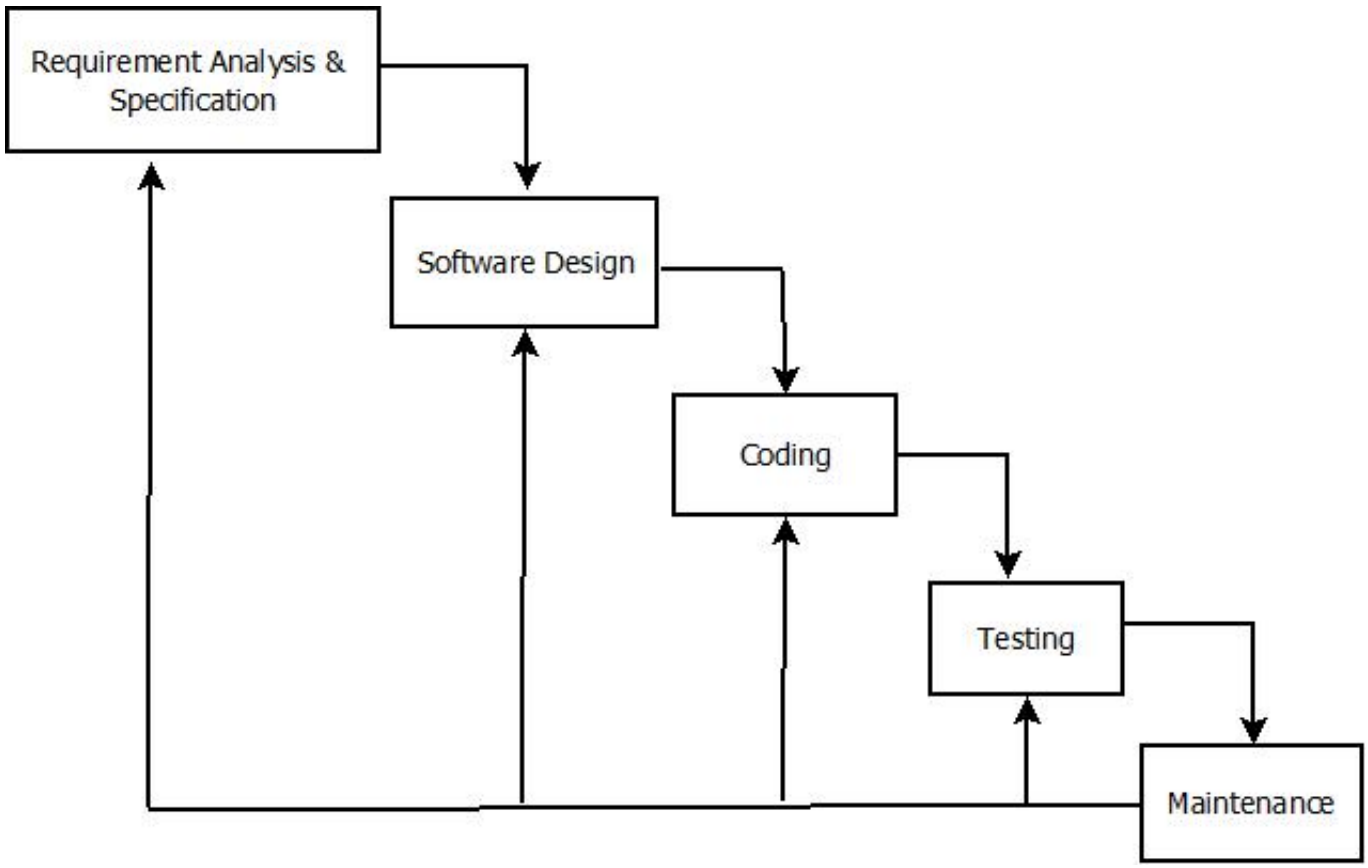
නිපදවන Software එකේ ස්වභාවය අනුව ඒ සඳහා ඒ ඒModel සුදුසු විය හැක.
මෙහි Waterfall Model හා Prototyping Models විස්තර වශයෙන් විමසා බලමු.

### Waterfall Model:-

එකිනෙකට වෙනස් තනි පියවරවලින් සෑදුම්ලත් රේඛීය අනුක්‍රමික Model 1කි. එනම් එක් පියවරක් අවසාන කිරීමෙන් පසු අනෙක් පියවර ආරම්භ කළ යුතුය. මෙහි ප්‍රධාන පියවර 5කි. ඒවා නම්,

1. Requirement Analysis & Specification
2. Software Design
3. Coding
4. Testing

5. Maintenance




## 1. Requirement Analysis & Specification:-

ගනුදෙනුකරුගෙන් තොරතුරු විමසීම හරහාSoftware 1න් සිදු විය යුතු සේවාවන්, එහි සීමාවන් (Constraints) හා එහි අරමුණු සනාථ කර ගනු ලැබේ. මෙහිදී Software එකෙහි Domain 1 හා Functionality, Behavior, Performance, Interface, Security ආදි වූ සියළුම අවශ්‍යතාවන් (Requirements) අවබෝධ කර ගැනීම සිදු විය හැක. පසුව මෙම අවශ්‍යතා, ගනුදෙනුකරුවන් (Users) හා Development Staff යන දෙපක්ෂයටම අවබෝධ කර ගත හැකි ආකාරයකට ඉදිරිපත් කෙරේ.

## 2. Software Design:-

Design ක්‍රියාවලිය මගින් පෙර පියවරේදී හඳුනා ගත් Requirements, Software Tools උපයෝගී කර ගනිමින් ක්‍රියාවට නැංවිය හැකි Software 1ක නිරූපණයක් බවට පරිවර්තනය කරයි. Design ක්‍රියාවලියේ ප්‍රධානම අභිප්‍රායන් වනුයේ Software Components (කොටස්), Software Architecture, Interfaces, Data Structures හා Algorithms හඳුනා ගැනීමයි.

## 3. Coding (Implementation) :-

පෙර පියවරේදී හඳුනා ගත් Design 1 යන්ත්‍රයට කියවා ගත හැකි (Machine Readable) ආකාරයකට පරිවර්තනය කළ යුතුය. මෙම පියවරේදී Software Design 1, Program හෝ Program Unit කාණ්ඩයක් ලෙස ලබා ගැනේ.Software Develop කිරීම සඳහා Programming Languages හෝ CASE Tools භාවිතා කළ හැක.Software 1 සත්‍ය වශයෙන්ම Implement කෙරෙන අවධිය මෙයයි.

## 4. Testing:-

Testing ක්‍රියාවලිය මගින්Software 1 නිවැරදිව වැඩ කරන බවත් පෙර හඳුනා ගත් Requirements සියල්ල තෘප්ත කරන බවත් සහතික කළ යුතුය. පරීක්ෂා කිරීමෙන් (Testing) පසු Software 1 ගනුදෙනුකාර පාර්ශවය වෙත නිකුත් කෙරේ.

## 5. Maintenance:-

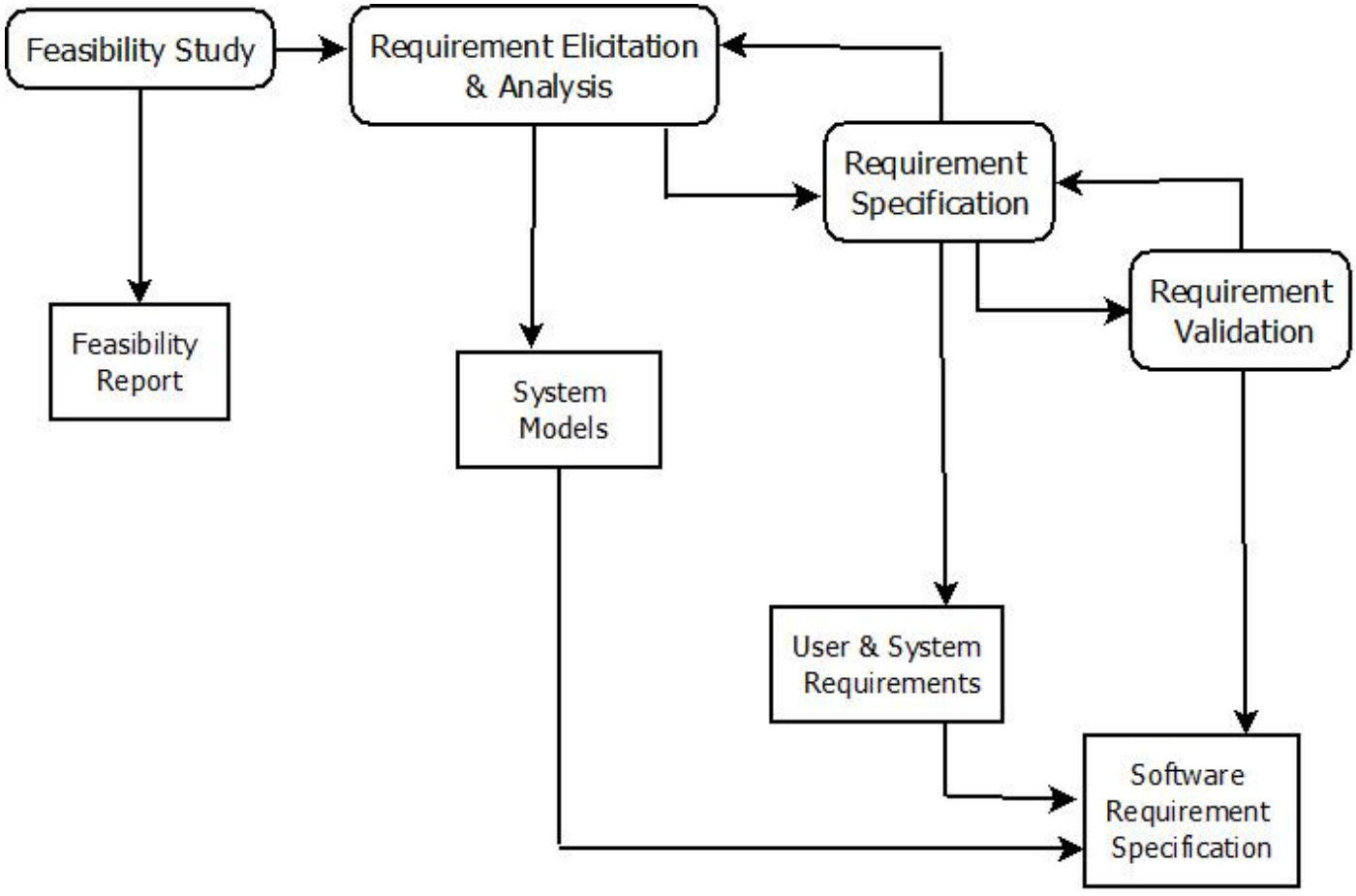
Software 1ගනුදෙනුකාර පාර්ශවය වෙත නිකුත් කිරීමෙන් අනතුරුවද නිශ්චිතවම වෙනස්කම්වලට භාජනය වේ. එහි පවතින දුබලතා නිවැරදි කොට නව අවශ්‍යතාවලට (Requirements) ගැලපෙන පරිදි නවීකරණය කළ යුතුය.

මෙම Model 1 භාවිතා කිරීමට නම්අවශ්‍යතාවන් (Requirements) සියල්ල පැහැදිලි විය යුතුය. එක් පියවරක් නිම කිරීමෙන් පසු නැවත වෙනස්කම් සිදු කළ නොහැක. උදාහරණයක් ලෙසDesign 1 නිම කිරීමෙන් පසුDesign 1 නැවත වෙනස් කළ නොහැක. මේ නිසා Software Development Process 1 අතරතුරදී Requirements වෙනස් වන්නේ නම් මෙමModel 1 සුදුසු නැත. එනම් Waterfall Model 1 වඩාත්ම සුදුසු වනුයේ Requirements පැහැදිලි හා ස්ථාවර අවස්ථා වලදීය.

## Waterfall Model සම්බන්ධ ගැටළු:-

- සත්‍ය වශයෙන්ම මෘදුකාංග ව්‍යාපෘති මෙවැනි අනුක්‍රමික ප්‍රවාහයක් අනුගමනය කරන්නේ ඉතාම සුළු වශයෙන් වීම.
- බොහෝ අවස්ථාවලදී සියළුRequirements පැහැදිලිව, සවිස්තරව ඉදිරිපත් කිරීමට ගනුදෙනුකරුවන් අපොහොසත් වීම.
- ගනුදෙනුකරුවන් ඉවසීමෙන් බලා සිටිය යුතුය. ක්‍රියාත්මක කළ හැකි Software 1ක් ලැබෙන්නේ අවසාන පියවරයන් හිදීය.

Enhanced Version of Waterfall Model




## Prototyping Models :-

ගනුදෙනුකරුවන් හට නව මෘදුකාංග පද්ධතිය තමන්ගේ වැඩ කටයුතු සඳහා යොදා ගන්නා ආකාරය කලින්ම සිතීමට අපහසුය.

විශේෂයෙන්ම ඉතා විශාල හා සංකීර්ණ පද්ධතියක් නම් එය නිපදවා භාවිතයට ගන්නා තෙක්ම අපහසු විය හැක.
මේ නිසා මෙම අපැහැදිලි Requirements පැහැදිලිව හඳුනා ගැනීමටPrototype 1ක් (මෘදුකාංගයේ මූලික අවස්ථාවක්) යොදා ගත හැක. මේ සඳහා ගනුදෙනුකරුගේ සහභාගීත්වයද ඇතිව මෙමPrototype 1 විනිශ්චය කළ යුතුය.
Prototyping හි ප්‍රධාන ශිල්ප ක්‍රම 2කි. ඒවා නම්,

1. Throw-away Prototyping
2. Evolutionary Prototyping

## 1. Throw-away Prototyping:-

මෙහි ප්‍රධාන පරමාර්ථය වනුයේ Requirements පැහැදිලිව අවබෝධ කර ගැනීමයි. මෙහිදී ව්‍යාපෘතිය ආරම්භ කෙරෙනුයේ දුබල ලෙස හඳුනා ගත් Requirements වලිනි. නමුත් Requirements පැහැදිලි කර ගත් වහාම Prototype 1 ඉවත් කර System 1 මුල සිටම Develop කෙරේ. නමුත් Prototype 1 සම්පූර්ණයෙන් ඉවත් කළ යුතු නැත. එහි අවසන් System 1 සඳහා භාවිතා කළ හැකි කොටස් වේ නම් ඒවා භාවිතා කළ හැක.

මෙහිදීRequirements හඳුනා ගත් පසු අප කටයුතු කරන්නේ Waterfall Model 1 ආකාරයටය.Prototype 1 ද ඉවත් කරයි. ඒ නිසාRequirements පසුව වෙනස්කම්වලට භාජනය විය නොහැක.
එනම්Throw-away Prototyping Model 1 වඩාත් සුදුසු වන්නේRequirements අපැහැදිලි නමුත් ස්ථාවර අවස්ථාවලදීය.

## Throw-away Prototyping සම්බන්ධ ගැටළු:-

ක්ෂණිකව Implement කිරීමේදී ඉතා වැදගත් ලක්ෂණPrototype 1න් මග හැරී යා හැක. කෙසේ වුවත්System එකෙහි සුරක්ෂිත බව වැනි වැදගත් කොටස්Prototype 1කට නැංවීම අපහසු විය හැක.
මෙහිදී Implement කිරීමක් සිදු වුවද ඒ සඳහා ගනුදෙනුකරු හා Developer අතර නීතිමය වලංගු භාවයක් නොමැත. Implement කිරීමෙන් පසු ගනුදෙනුකරුPrototype 1 ප්‍රතික්ෂේප කිරීමේ අවදානමක් ඇත.
Prototype Implementation 1කදී Reliability, Robustness වැනි ගුණාංග ප්‍රමාණවත් ලෙස පරීක්ෂාවට ලක් කළ නොහැක.

## 2. Evolutionary Prototyping:-

මෙහිදී ද පෙර පරිදීම Prototype 1ක් භාවිතයෙන් Requirements පැහැදිලි කර ගනු ලැබේ. ඉන්පසුPrototype 1 ඉවත් නොකරRequirements තෘප්ත වන තෙක් එයම වැඩි දියුණු කරයි. මෙහිදීRequirements වල වෙනස් වීම් වලට ඉඩ දිය හැක. වෙනස් වනRequirements වලට අනුවPrototype 1 ද වැඩි දියුණු කළ යුතුය.System 1 සතුටුදායක මට්ටමකට පැමිණි විට ගනුදෙනුකරු වෙත නිකුත් කළ හැක.
Evolutionary PrototypingModel 1 වඩාත් සුදුසු වන්නේRequirements අපැහැදිලි හා අස්ථාවර අවස්ථාවලදීය.

## Evolutionary Prototyping හි වාසි :-

- Prototype 1 සඳහා දැරූ උත්සාහය අපතේ නොයාම.
- Waterfall Model 1ට වඩා වේගවත් වීම.
- ආරම්භයේ සිටම ගනුදෙනුකරුගේ සහභාගීත්වය ඉහළ මට්ටමක පැවතීම.
- තාක්ෂණික හෝ වෙනත් ගැටළු ඉක්මනින් හඳුනා ගන්නා හෙයින් අවදානම අඩු වේ.

## Evolutionary Prototyping හි අවාසි :-

- Prototype 1 ඉක්මනින් සකස් කෙරෙන නිසා එතරම් හොඳ Documentation 1ක් සිදු නොවේ.
- දිගින් දිගටම කෙරෙන වෙනස්කම් නිසා Prototype 1හි ව්‍යුහයට හානි විය හැක. එමගින් නඩත්තු කිරීම අපහසු හා අධික වියදම් සහිත විය හැක.

- Prototype 1 සඳහා භාවිතා කරන Language 1 සෑම විටම අවසන්System 1 සඳහා සුදුසු නොවිය හැක.

මීට අමතරව ව්‍යාපෘතියේ ලක්ෂණ අනුව අනෙකුත් Software Process Model ද ඒ සඳහා සුදුසු විය හැක.

- කෙටි කලකින් නිම කළ යුතු ව්‍යාපෘති සඳහාRapid Application Development සුදුසු වේ.
- ඉහළ නිරවද්‍යතාවයක් අවශ්‍ය වන ව්‍යාපෘති සඳහා Formal Development සුදුසුය.
- අවදානම ඉහළ ව්‍යාපෘති සඳහාSpiral Model සුදුසුය.