भारतीय संस्कृति, शिक्षा और दर्शन

विभिन्न संकलपनाएं

सम्पादक : डॉ. दिनेश चन्द्र शर्मा

भारतीय संस्कृति, शिक्षा और दर्शन

विभिन्न संकल्पनाएँ

सम्पादक :

डॉ. दिनेश चंद्र शर्मा

Ph.D., D.Lit. (Sub.)

निदेशक, कॉम्पिटिशन प्लस ग्रुप ऑफ इन्स्टीट्यूशन,
अजमेर

Books n Books

51/1509, आचमन, वेदविहार,
लोहाखान, अजमेर-305001

प्रकाशक :
पुष्पा शर्मा
Books n Books
51/1509, आचमन, वेदविहार,
लोहाखान, अजमेर-305001
ई-मेल : bnbajmer@gmail.com

संस्करण
प्रथम 2025

ISBN : 978 - 81 - 993854 - 7 - 4

मूल्य - 300/-

रचना -
भारतीय संस्कृति, शिक्षा और दर्शन विभिन्न संकल्पनाएँ

सम्पादक -
डॉ. दिनेश चंद्र शर्मा

आवरण -
श्रीनिवास प्रिन्टोग्राफिक्स, अजमेर

मुद्रक -
श्री श्याम ऑफसेट, किशनगढ़

SOFTWARE ENGINEERING

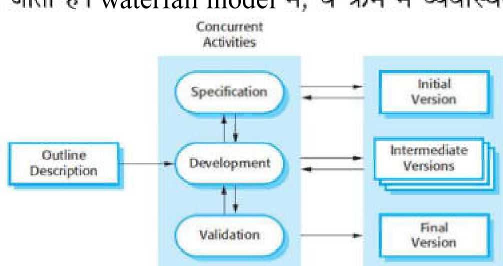
Shahin Parveen
Assistant Professor (Deptt. of Science and Technology)
Hukumchand Noble Institute of Science and Technology

Software :- Computer को operate करने और specific tasks को execute करने के लिए काम आने वाला set of instructions, data or programs है। यह hardware का opposite है, जो computer के physical aspects का वर्णन करता है। Software एक generic term है जिसका उपयोग किस device पर run होने वाले applications, scripts and programs को रेफर करने के लिए किया जाता है।

System Software एक कंप्यूटर प्रोग्राम का एक प्रकार है जिसे computer's hardware और application programs को run करने के लिए डिजाइन किया गया है। अगर हम computer System को एक स्तरित मॉडल के रूप में कल्पित करते हैं, तो System Software hardware और user applications के बीच का interface होगा।

Software Process :- Programs के रूप में निर्देशों का group जो software computer system को नियंत्रित करने और hardware components को संसाधित करने के लिए उपयोग किया जाता है। एक set of activities का उपयोग एक software product बनाने के लिए किया जाता है। इस set को एक software process कहा जाता है।

specification, development, validation and evolution की चार basic process activities अलग-अलग development processes में अलग-अलग प्रकार से व्यवस्थित किया जाता है। waterfall model में, वे क्रम में व्यवस्थित किए जाते



हैं, जबकि incremental development में वे अंत:स्थापित होते हैं।

Any software process shall comprise the following four activities :

A. Software specification (or requirements engineering) :- software की मुख्य कार्यात्मकताओं और उनके आसपास की बाधाओं को परिभाषित करें।

B. Software design and implementation :- software को designed and programmed किया जाना है।

C. Software verification and validation :- software को इसके विनिर्देशों (Specifications) के अनुरूप होना चाहिए और customer's needs को पूरा करना चाहिए।

D. Software evolution (software maintenance) :- customer and market की आवश्यकताओं में बदलाव को पूरा करने के लिए software को modified किया जा रहा है।

व्यवहार में, वे उप-गतिविधियों जैसे कि आवश्यकता requirements validation, architectural design, unit testing आदि शामिल हैं।

Waterfall Model :-

Waterfall Model (वॉटरफॉल मॉडल) SDLC(सिस्टम डेवलपमेंट लाइफ साइकिल) का एक famous और good version है। waterfall model (वॉटरफॉल मॉडल) linear तथा sequential मॉडल है इसका मतलब यह है कि एक phase of डेवलपमेंट तब तक शुरू नही हो सकता जब तक कि उसका पिछला वाला phase complete नही हो जाता है। हम वॉटरफॉल मॉडल में phases overlap नही कर सकते है।

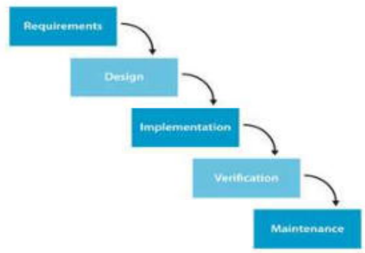
हम वॉटरफॉल को निम्न तरीके से imagine कर सकते है :-

एक बार जब पानी चट्टान के किनारे के ऊपर से प्रवाहित होने लगता है और पहाड़ के नीचे की ओर गिरने लगता है और यह पानी ऊपर की ओर वापस नही जा सकता है।

इसी तरह waterfall model भी काम करता है, एक बार डेवलपमेंट का एक phase पूरा हो जाता है तो हम अगले phase में जा जाते है लेकिन पिछले phase में वापस नही जा सकते हैं।

वॉटरफॉल मॉडल में, एक phase का आउटपुट दूसरे phase के लिए इनपुट की तरह काम करता है।

वॉटरफॉल मॉडल में निम्नलिखित 5 phases होते है (waterfall phases) :-



Waterfall model

1. Requirement phase :- requirement phase वॉटरफॉल मॉडल का पहला phase है। इस फेज में सिस्टम की requirements इकट्ठा तथा documented की जाती है। यह फेज बहुत महत्वपूर्ण होती है क्योंकि इसी फेज पर अगला phase depend करता है।

2. Design phase :- design phase इस तथ्य पर आधारित होता है कि सॉफ्टवेयर का निर्माण किस प्रकार होगा। डिजाइन फेज का मुख्य उद्देश्य सॉफ्टवेयर सिस्टम का blueprint तैयार करना है जिससे कि आने वाले phases पर किसी प्रकार की कोई दिक्कत का सामना ना करना पड़े और requirement phase में जो भी requirements है उनका solution निकाल लिया जाएँ।

3. Implementation phase :- इस प्रोसेस में हार्डवेयर, सॉफ्टवेयर तथा एप्लीकेशन प्रोग्राम्स को install किया जाता है तथा डेटाबेस डिजाइन को implement किया जाता है। उससे पहले की डेटाबेस डिजाइन को implement किया जाएं सॉफ्टवेयर को टेस्टिंग, कोडिंग तथा debugging प्रोसेस से होकर गुजरना पड़ता है। वॉटरफॉल में यह सबसे लंबे समय तक चलने वाला phase है।

4. Verification phase :- यहाँ सॉफ्टवेयर verify किया जाता है और यह check किया जाता है कि हमने सही product बनाया है। इस फेज में टेस्टिंग की अलग-अलग प्रकार की की जाती है तथा सॉफ्टवेयर के हर area की check की जाती है।

माना कि हमने सॉफ्टवेयर को ठीक verify नही किया और इसमें कोई defect बच जाता है तो इसका इस्तेमाल कोई भी नही करेगा इसलिए verification

बहुत जरूरी है।

verification का एक फायदा यह है कि इससे सॉफ्टवेयर के fail होने का खतरा कम हो जाती है।

5. Maintenance phase :- यह वॉटरफॉल का सबसे अंतिम phase है। जब सिस्टम बनके तैयार हो जाता है तथा यूजर उसका प्रयोग करना शुरू कर देते है तब जो problems उसमें आती है उनको time-to-time हल करना पड़ता है। तैयार सॉफ्टवेयर को समय अनुसार उसका ख्याल रखना तथा उसे maintain रखना ही maintenance कहलाता है। SDLC में तीन प्रकार के maintenance होते है :-

A. Corrective maintenance

B. Adaptive maintenance

C. Perfective maintenance.

Waterfall Model का use कब किया जाता है :

• इस model का use तब किया जाता है जब user की जो requirements होती है वो fixed और clear हो जिन्हें change नहीं किया जा सकता है।

• इसका उपयोग small projects के लिए ज्यादातर किया जाता है।

• अगर आवश्यक विशेषज्ञता वाले पर्याप्त संशाधन स्वतंत्र रूप से उपलब्ध हो।

• Technology को अच्छी तरह से समझ लिया हो तो इस मॉडल का उपयोग कर सकते हैं।

• अगर ऐसी स्थिती है जहां आप इस मॉडल का उपयोग कर सकते है जब ह्द्व्ह्य्द्म बिल्कुल न हो या minimum हो।

• क्लाइंट की कोई अनिश्चित आवश्यकताएं (ambiguous requirements) नहीं हो।

Waterfall Model के Advantages :

1. यह मॉडल सरल और समझने में आसान है।

2. यह छोटे प्रोजेक्ट्स के लिये बेहद उपयोगी है।

3. इस मॉडल को मैनेज करना आसान है।

4. अंतिम लक्ष्य को जल्दी निर्धारित किया जाता है।

5. इस मॉडल के प्रत्येक phase को अच्छी तरह से स्पष्ट किया गया है।

6. यह चीजों को करने के लिये एक स्ट्रक्चर तरीका है।

7. यह एक base model है इसके बाद जितने भी SDLC models आये वो इसी को देखते हुए बनाये गए थे हालांकि उन्होंने इसकी खामियों को दूर करने का

काम किया।

8. इस मॉडल में पहले phase के सफलतापूर्वक पूरा हो जाने के बाद ही हम अगले phase में जा सकते है ताकि phases के बीच कोई overlapping न हो।

Waterfall Model के Disadvantages :

1. Opaque (अस्पष्ट)

2. इस मॉडल में development process के आरंभ में complete और accurate requirements की अपेक्षा की जाती है।

3. Development life cycle के दौरान बहुत देर तक working s/w उपलब्ध नहीं होता है।

4. हम पिछले phase में वापस नहीं जा सकते जिसके कारण आवश्कताओं में बदलाव करना बहुत मुश्किल होता है।

5. इसमें जोखिम का आंकलन नहीं किया जाता इसलिए इस मॉडल में high risk और uncertainty होती है।

6. यहाँ testing period बहुत पीछे आता है।

7. वहाँ अपने सीक्वेंशियल नेचर के कारण यह मॉडल यह दुनिया वर्तमान में realistic नहीं है।

8. यह बड़े और जटिल प्रोजेक्ट्स के लिए एक अच्छा मॉडल नहीं है।

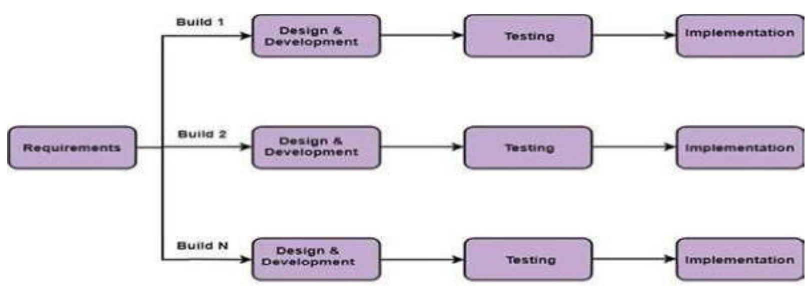


Fig: Incremental Model

Incremental Model :- Incremental Model (वृद्धिशील मॉडल) में software development process को कई increments में बांटा जाता है और प्रत्येक increment में एक समान phases को follow किया जाता है। समझे जाएँ तो इस मॉडल के तहत एक बड़े project को कई modules या builds में विकसित करते हैं।

For example हम customer की requirements को एकत्रित करते है, अब

पूरा software एक बार मे बना देने के बजाय हम पहले कुछ requirements को लेते है और उनके आधार पर software का एक module या function बनाकर customer को deliver करते हैं। फिर हम कुछ और requirements लेते है और उनके आधार पर एक और module उस software में add करते है।

Such that every increment पर module software में add किया जाता है जब complete system नहीं बन जाता। इसके बावजूद एक बड़े projects को multiple iterations/parts में बनाने के लिए जो requirements हैं वो clear होनी चाहिए।

यदि Incremental methodology के पूरे सिद्धांत को समझें तो इसमें शुरुआत एक initial implementation को develop करके की जाती है, फिर उस पर user के feedback लिए जाते है और जब तक एक accepted system विकसित नहीं हो जाता इसे कई versions के माध्यम से develop किया जाता है। शुरुआती iterations में software की महत्वपूर्ण functionalities को विकसित किया जाता है।

Software module की every successive release previous release में function add करती है। यह प्रक्रिया तब तक चलती है जब तक final software नहीं प्राप्त हो जाता।

Incremental Model किसका उपयोग करना चाहिए :

- इस मॉडल का उपयोग तब करना चाहिए जब system requirements को अच्छे से समझा जा चुका हो।
- जब software के early release की मांग हो।
- जब सॉफ्टवेयर इंजीनियरिंग टीम बहुत बेहतर तरीके से कुशल या प्रशिक्षित नहीं हो।
- जब Large-sized projects हो।
- जब new technology को use किया जा रहा हो।
- जब high risk features और goals शामिल हो।

Phases of Incremental Model :-

Communication :- पहले phase में हम customer के साथ आमने-सामने बातचीत करके उसकी जो mandatory requirements होती है उन्हें collect करते है। जैसे कि customer को उसके software में कौन-कौन सी functionalities चाहिये, आदि।

Planning :- इसमें requirements को various modules में divide किया जाता है और उनके आधार पर planning की जाती है।

Modeling :- इस स्टेज में हर एक module का design तैयार किया जाता है। design तैयार हो जाने के पश्चात् हम कई modules के बीच एक specific module को लेते है और उसे DDS (Design Document Specification) में save करते है। इस document में ERDs और DFDs जैसे diagrams को शामिल किया जाता है।

Construction :- यहाँ हम उस specific module के design के आधार पर construction शुरू करते हैं। अर्थात module का design coding में implement किया जाता है। एक बार code लिखने के बाद उसका test किया जाता है।

Deployment :- code की testing पूरी हो जाने के बाद अगर वह module सही तरह से work कर रहा हो तो उसे customer को use करने के लिए दे दिया जाता है।

इसके बाद next module को same phases के माध्यम से develop किया जाता और उसे previous module के साथ combine कर दिया जाता है। इससे customer के लिए नई functionality उपलब्ध हो जाती है। ऐसा तब तक किया जाता है जब तक complete modules विकसित नहीं किये जाते।

Incremental Model के प्रकार

1. Staged Delivery Model :-

Incremental model के यह रूप में project का एक part एक समय में develop किया जाता है। हम यह भी कह सकते हैं कि project के parts एक-एक करके develop किए जाते हैं। इसमें हम customer को पूरा software एक बार में नहीं बल्कि कई versions में deliver करते हैं।

2. Parallel Development Model :-

इसमें एक ही समय मे विभिन्न modules को develop किया जाता है। प्रत्येक module के लिए एक अलग team होती है जो parallelly प्रोजेक्ट में काम कर रही होती है। यह development में लगने वाले समय को कम करता है, i.e. TTM (Time to Market).

Advantage of Incremental Model

1. महत्वपूर्ण modules/functions को पहले develop किया जाता है और फिर शेष को chunks में add किया जाता है।

2. Software development life cycle (SDLC) के समय working s/w को quickly और early तैयार किया जाता है।

3. यह मॉडल requirements और scope को change करने के लिये flex-

ible और less expensive है।

4. प्रत्येक build पर customer respond कर सकता है और यदि किसी changes की आवश्यकता हो तो feedback प्रदान कर सकता है।

5. Project की progress मापी जा सकती है।

6. एक छोटे iteration के समय test और debug करना आसान है।

7. it is easy to identify Errors.

Disadvantage of Incremental Model

1. Management एक ongoing process है जिसको handled करना पर्याप्त है।

2. बतौरपूर्व project को breaking into pieces, incrementally built किया जा सके।

3. software की complete requirements clear होने चाहिये।

4. इसके लिए better planning और डिजाइनिंग की आवश्यकता होती है।

5. इस मॉडल में काम लगने वाली total cost higher होती है।

Rational Unified Process (RUP) :-

Rational Unified Process (RUP) object-oriented models के लिए एक सॉफ्टवेयर विकास प्रक्रिया है। इसे एकीकृत प्रक्रिया मॉडल के रूप में भी जाना जाता है। यह रैशनल कॉरपोरेशन द्वारा बनाया गया है और इसे UML (Unified Modeling Language) का उपयोग करके डिजाइन और प्रलेखित किया गया है। यह प्रक्रिया IBM रैशनल मेथड कंपोजर (RMC) उत्पाद में शामिल है। आईबीएम (इंटरनेशनल बिजनेस मशीन कॉर्पोरेशन) हमें एकीकृत प्रक्रिया को अनुकूलित, डिजाइन और वैयक्तिकृत करने की अनुमति देता है। आर.यू.पी. का प्रस्ताव इवर जैकबसन, ग्रेडी बूटच और जेम्स रामबॉघ द्वारा किया गया है। आरयूपी की कुछ विशेषताओं में उपयोग-केस संचालित, पुनरावृत्त (प्रक्रिया की पुनरावृत्ति) और प्रकृति द्वारा वृद्धिशील (मूल्य में वृद्धि), वेब तकनीक का उपयोग करके ऑनलाइन वितरित, मॉड्यूलर और इलेक्ट्रॉनिक रूप में अनुकूलित या सिलवाया जा सकता है, आदि शामिल हैं। आरयूपी अप्रत्याशित को कम करता है विकास लागत और संसाधनों की बर्बादी को रोकता है।

1. शुरुआत (Inception) :-

• संचार एवं नियोजन प्रमुख हैं।

• परियोजना के दायरे की पहचान करने के लिए उपयोग-केस मॉडल का व्यवहार करता है जिससे प्रबंधकों को लागत और आवश्यक समय का अनुमान लगा पाने में सहायता मिलती है।

• ग्राहकों की आवश्यकताओं की पहचान की जाती है और परियोजना के लिए योजना बनाना आसान हो जाता है।

• परियोजना योजना, परियोजना लक्ष्य, जोखिम, उपयोग-केस मॉडल और परियोजना विवरण तैयार किया जाता है।

• परियोजना को मील के पत्थर के मानदंडों के अनुसार जाँचा जाता है और यदि यह इन मानदंडों को पारित नहीं कर पाता है तो परियोजना को रद्द किया जा सकता है या फिर से डिजाइन किया जा सकता है।

2. विस्तार (Elaboration) :-

• इसमें प्लानिंग और मॉडलिंग प्रमुख हैं।

• एक विस्तृत मूल्यांकन और विकास योजना लागू की जाती है और जोखिम कम हो जाते हैं।

• उपयोग-मामले मॉडल (लगभग 80 प्रतिशत), व्यावसायिक मामले और जोखिमों को संशोधित या पुन: परिभाषित करें।

• फिर से, मील के पत्थर मानदंडों के खिलाफ जांच की गई और यदि यह इन मानदंडों को पारित नहीं कर सका तो परियोजना को फिर से रद्द किया जा सकता है या फिर से डिजाइन किया जा सकता है।

• निष्पादन योग्य आर्किटेक्चर बेसलाइन।

3. Construction :-

• परियोजना विकसित और पूर्ण हो गई है।

• सिस्टम या सोर्स कोड तैयार किया जाता है और फिर परीक्षित किया जाता है।

• कोडिंग की जाती है।

4. संक्रमण (Transition) :-

• Ultimate project general public के लिए रिलीज की जाती है।

• project को विकास से प्रोडक्शन में transfer करें।

• pdate project documentation

• बीटा टेस्ट आयोजित किया जाता है।

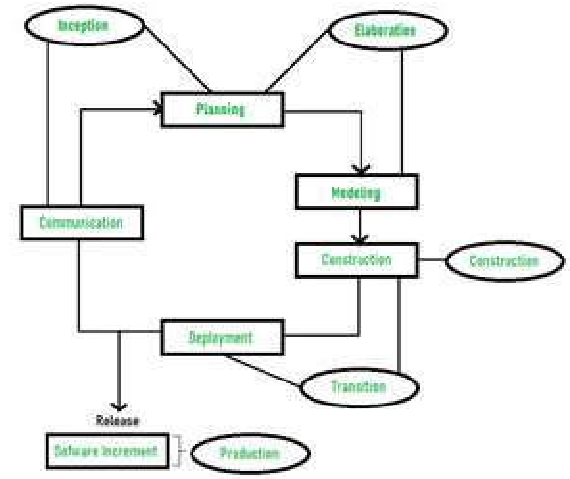
• जनता से मिले फीडबैक के आधार पर प्रोजेक्ट से खामियां दूर की जाती हैं।

5. उत्पादन (Production) :-

• मॉडल का आखिरी चरण।

• परियोजना को तदनुसार बनाए रखा और अद्यतन किया जाता है।

Phases of RUP :- There is total of five phases of the life cycle of



Advantages :-

1. यह अच्छा दस्तावेजीकरण प्रदान करता है, यह प्रक्रिया को स्वयं पूरा करता है।
2. यह जोखिम-प्रबंधन सहायता प्रदान करता है।
3. यह घटकों का पुनः उपयोग करता है और इसलिए कुल समय अवधि कम है।
4. अच्छी ऑनलाइन समर्थन के रूप में ट्यूटोरियल और प्रशिक्षण उपलब्ध है।

Disadvantages :-

1. विशेषज्ञ पेशेवर की टीम की आवश्यकता है, क्योंकि प्रक्रिया जटिल है।
2. जटिल एवं समुचित रूप से व्यवस्थित न होने वाली प्रक्रिया।
3. जोखिम प्रबंधन पर अधिक निर्भरता।
4. बार-बार एकीकृत करना कठिन।

Agile model :-

Agile model जो है वह iterative तथा incremental models का एक combination है अर्थात् यह iterative तथा incremental models से मिलकर बना है।

Agile model में process flexibility और customer satisfaction को Arior ARIORITYAANYA दिया जाता है।

Earlier times में iterative waterfall model का उपयोग सॉफ्टवेयर बनाने

के लिए किया जाता था। लेकिन आज का समय है तो developers को बहुत सारी मुश्किलों का सामना करना पड़ता है. सबसे बड़ी मुश्किल यह है कि software development के अकेले customer सॉफ्टवेयर में changes करने को कहता है। इन changes को करने में बहुत बहुत सारा समय निर्वाह अत: पैसे लगते हैं, तो इन सब कमीयों को पूरा करने के लिए 1990 के दशक में agile model को प्रस्तावित किया गया।

Agile model को मुख्यतया software development के बीच में changes करने के लिए ही बनाया गया था जिससे कि सॉफ्टवेयर प्रोजेक्ट को जल्दी से पूरा किया जा सके।

Agile model में निम्नलिखित steps होते है :-

1. Requirement gathering
2. Requirement analysis
3. Design
4. Coding
5. nit testing
6. Acceptance testing

Agile Model में software product को smaller incremental parts में divided किया जाता है। इसमें पहले सबसे छोटे part को developed किया जाता है और फिर उससे बड़ा और हर incremental part को iteration पर developed किया जाता है।

प्रत्येक iteration को छोटा रखा जाता है जिससे कि उसे आसानी से manage किया जा सके तथा उसे दो तीन हफ्तों में पूरा किया जा सके। एक समय में केवल एक ही iteration को plan, develop तथा deploy किया जाता है।

Principles of Agile Model

1. Software development के दौरान customer के साथ contact रखे रखने के लिए तथा requirement समझने के लिए development team में एक customer representative होता है। जब कोई iteration पूरा कर लिया जाता है तो stakeholders तथा customer representative उसे review करते है तथा requirements को दूबारा evaluate करते है।

2. Customer की requirements का पालन करने के लिए working software का demo दिया जाता है। अर्थात् इसमें सिर्फ documentation पर depend नहीं रहते है।

3. सॉफ्टवेयर के incremental versions को कुछ हफ्तों बाद customer representative को deliver करना पड़ता है।

4. इस मॉडल में यह recommended किया जाता है कि development team का size छोटा (5 से 9 व्यक्ति) होना चाहिए ताकि team member जो है वह face to face communicate कर सकें।

5. Agile model इस बात पर emphasis करता है कि सॉफ्टवेयर में जब भी कोई changes करने हो तो उसे तेजी से पूरा कर लिया जाए।

6. Agile development में दो programmers एक साथ काम करते है. एक programmer कोडिंग करता है तो दूसरा उस code को review करता है. दोनों programmers अपने कामों को बदलते रहते है यानी कि कभी कोई coding करता है तो कभी कोई review।

Advantage

1. इसमें दो programmers एक साथ काम करते है जिससे coding बहुत ही अच्छी होती है तथा उसमें बहुत ही कम गलतियाँ होती है।

2. इसमें software project को बहुत कम समय में पूरा कर लिया जाता है।

3. इसमें customer representative को प्रत्येक iteration का idea होता है जिससे वह requirement को आसानी से change कर सकते है।

4. यह सॉफ्टवेर development की बहुत ही वास्तविक approach है।

5. इसमें teamwork पर ध्यान दिया जाता है।

6. इसमें rules बहुत कम होते है और documentation भी ना के बराबर होता है।

7. यहाँ planning की आवश्यकता नहीं पड़ती।

8. यह आसानी से manage हो सकता है।

9. यह developers को flexibility प्रदान करता है।

Disadvantage

1. यह complex dependencies का handling नहीं कर सकता है।

2. इसमें formal documentation की कमी के कारण development में धुंधलापन हो जाता है।

3. यह काफी customer representative पर निर्भर रहता है अगर customer representative गलत information दे देते है तो सॉफ्टवेयर गलत हो सकता है.

4. इसमें only experienced programmers ही कोई decision ले सकते है. नए programmers कोई भी decision नहीं ले सकते है।

5. इसमें software development के शुरुवात में सॉफ्टवेयर को बनाने में कितना effort तथा time लगेगा का पता नहीं लग पाता है।

❑❑❑