के लिए एक प्रयोगशाला मैनुअल

सॉफ़्टवेयर इंजीनियरी
(3161605)

होना। सेमेस्टर 6 (सूचना प्रौद्योगिकी)

तकनीकी शिक्षा निदेशालय, गांधी नगर,
गुजरात ।

सरकारी इंजीनियरिंग कॉलेज, भावनगर
प्रमाणपत्र

यह प्रमाणित करना है कि Mr./ms। _____ _____
नामांकन संख्या _____ बी.ई. सेमेस्टर ____ जानकारी
इस संस्थान की प्रौद्योगिकी (GTU कोड: ___) ने संतोषजनक रूप से कॉम को पलेट किया है
व्यावहारिक / ट्यूटोरियल डब्ल्यूविषय सॉफ्टवेयर इंजीनियरिंग के लिए ऑर्क (3161605)
शैक्षणिक वर्ष 202 3-24 के लिए।

जगह: _____
तारीख: _____

संकाय सदस्य का नाम और संकेत

विभाग प्रमुख

प्रस्तावना
किसी भी प्रयोगशाला/व्यावहारिक/क्षेत्र के काम का एम एइन आदर्श वाक्य आवश्यक कौशल बढ़ाने के लिए है एक एनडी क्रिएट ई क्षमता
छात्रों के बीच में प्रासंगिक दक्षताओं को विकसित करके वास्तविक समय की समस्या को हल करने के लिए साइकोमोटर डोमेन। इसे ध्यान में रखते हुए, GTU ने एक योग्यता तैयार की है -केंद्रित परिणाम -

इंजीनियरिंग डिग्री कार्यक्रमों के लिए आधारित पाठ्यक्रम जहां पर्याप्त वेटेज व्यावहारिक दिया जाता है
काम। यह छात्रों के बीच कौशल में वृद्धि के महत्व को दर्शाता है, और यह ध्यान देता है
छात्रों, प्रशिक्षकों और संकाय के बीच व्यावहारिक एस के लिए आवंटित हर समय का उपयोग करना
सदस्य केवल अध्ययन -टाइप के बजाय प्रयोगों का प्रदर्शन करके प्रासंगिक परिणामों को प्राप्त करने के लिए
प्रयोग। यह एक योग्यता -foc के प्रभावी कार्यान्वयन के लिए जरूरी हैउपयोग किए गए परिणाम -आधारित
पाठ्यक्रम कि प्रत्येक व्यावहारिक को प्रासंगिक विकसित करने और बढ़ाने के लिए एक उपकरण के रूप में
सेवा करने के लिए डिज़ाइन किया गया है
प्रत्येक छात्र के बीच विभिन्न उद्योग द्वारा आवश्यक योग्यता। ये साइकोमोटर कौशल बहुत हैं
कक्षा में पारंपरिक चाक -और -बोर्ड सामग्री वितरण विधि के माध्यम से विकसित करना मुश्किल है।
तदनुसार, इस लैब मैनुअल को उद्योग पर ध्यान केंद्रित करने के लिए डिज़ाइन किया गया है।
अवधारणा एस और थियो साबित करने के लिए व्यावहारिक एस का संचालन करने की पुरानी प्रथाr ies।
इस लैब मैनुअल का उपयोग करके, छात्र पहले से पहले प्रासंगिक सिद्धांत और प्रक्रिया से गुजर सकते हैं
वास्तविक प्रदर्शन, जो रुचि पैदा करता है, और छात्रों को इससे पहले एक मूल विचार हो सकता है
प्रदर्शन। यह, बदले में, छात्रों के बीच पूर्व -निर्धारित परिणामों को बढ़ाता है। में प्रत्येक प्रयोग
यह मैनुअल योग्यता, उद्योग -प्रासंगिक कौशल, पाठ्यक्रम परिणामों के साथ -साथ व्यावहारिक के साथ शुरू
होता है
परिणाम (उद्देश्य)। छात्र सुरक्षा और आवश्यक एहतियात भी प्राप्त करेंगेएस लिया जाना है
व्यावहारिक प्रदर्शन करते समय।
यह मैनुअल छात्र -सेंट्रिक लैब गतिविधियों को सुविधाजनक बनाने के लिए संकाय सदस्यों को दिशानिर्देश भी
प्रदान करता है
प्रत्येक प्रयोग के माध्यम से आवश्यक संसाधनों की व्यवस्था और प्रबंधन करके कि छात्रों को
परिणामों को प्राप्त करने के लिए आवश्यक सुरक्षा और एनईसी निबंध सावधानियों के साथ प्रक्रियाओं का
पालन करें। यह भी
रूब्रिक्स प्रदान करके छात्रों का मूल्यांकन कैसे किया जाएगा, इसका अंदाजा है।
सॉफ्टवेयर इंजीनियरिंग एक व्यवस्थित, परिभाषित और मापक का एक अनुप्रयोग हैई दृष्टिकोण जो शुरू
होता है
आवश्यकता के साथ विनिर्देशन और प्रगति की योजना, मॉडलिंग और परीक्षण के साथ प्रगति, और एस का
समापन
तैनाती के साथ। यह एक स्तरित प्रतिमान है जिसमें Proces SES, तरीके और उपकरण शामिल हैं
गुणवत्ता फोकस का आधार। सॉफ्टवेयर इंजीनियरिंग दृष्टिकोण का मुख्य उद्देश्य प्रतिबद्ध है
अधिक गुणवत्ता के साथ निर्धारित समय और बजट के भीतर सॉफ्टवेयर उत्पादों को विकसित करना।
गुणवत्ता
उत्पाद दृढ़ता, वस्तु और प्रसन्नता को प्रेरित करता है।
अत्यधिक ध्यान रखा गया हैn इस लैब मैनुअल को तैयार करते समय; हालांकि, वहाँ एक मौका है
सुधार। इसलिए, हम सुधार और हटाने के लिए रचनात्मक सुझावों का स्वागत करते हैं
त्रुटियां, यदि कोई हो।
कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - पाठ्यक्रम परिणाम मैट्रिक्स

पाठ्यक्रम परिणाम (COS):
सह -1: SRS (सॉफ्टवेयर आवश्यकता विनिर्देश) दस्तावेज़ और SPMP (सॉफ्टवेयर प्रोजेक्ट) तैयार करें प्रबंधन योजना) दस्तावेज़।
सह -2: कार्यात्मक उन्मुख और ऑब्जेक्ट -ऑब्जेक्टेड दृष्टिकोण की अवधारणा को लागू करेंसॉफ्टवेयर डिजाइन।
सह -3। पहचानें कि सॉफ्टवेयर उत्पाद की गुणवत्ता को कैसे सुनिश्चित करें, विभिन्न गुणवत्ता मानकों, और सॉफ्टवेयर समीक्षा तकनीक।
सह -4। विभिन्न परीक्षण तकनीकों और परीक्षण योजना को लागू करें।
सह -5। आधुनिक चुस्त विकास के तहत सक्षम
सीनियर
सं।
1। विभिन्न प्रकार के सॉफ्टवेयर प्रक्रिया मॉडल के साथ अध्ययन
तुलना करें और पता करें कि कौन सा प्रक्रिया मॉडल होगा
आपकी चयनित परियोजना के लिए उपयुक्त है। √
2। चर्चा करेंप्रोजेक्ट मैनेजमेंट: प्रोज सीटी प्लानिंग और
अपनी परियोजना के बारे में परियोजना का निर्धारण। √ √
3। सॉफ्टवेयर आवश्यकता विनिर्देश (एसआरएस) तैयार करें
चयनित परियोजना के लिए दस्तावेज़। √ √
4।
अपने चयनित परियोजना के लिए डेटा प्रवाह आरेख बनाएं। √ √
5। अपने चयनित के लिए इकाई -Relationship आरेख बनाएं
प्रोजेक्ट of
6। अपने चयनित परियोजना के लिए USECASE आरेख बनाएं। √ √
7। बेसिक कोकोमो मॉडल को लागू करके समस्या को हल करें। √ √
8। मॉडलिंग यूएमएल क्लास आरेख और सेक्ence आरेख √ √
9। परीक्षण करने के लिए विभिन्न परीक्षण मामलों को डिजाइन करें
सिस्टम और विभिन्न प्रकार के परीक्षण भी करते हैं। √ √
10। के लिए किसी भी दो खुले स्रोत उपकरणों का अध्ययन
इन्फ्रास्ट्रक्चर ऑटोमा टियोन, कॉन्फ़िगरेशन प्रबंधन
परिनियोजन स्वचालन, प्रदर्शन एनसीई प्रबंधन, लॉग
प्रबंधन की निगरानी। √

कोमल
सॉफ्टवेयर इंजीनियरिंग

उद्योग प्रासंगिक कौशल
निम्नलिखित उद्योग प्रासंगिक योग्यता स्टड में विकसित होने की उम्मीद हैमें प्रवेश करना
इस प्रयोगशाला का व्यावहारिक कार्य करना।
1। सॉफ्टवेयर के विकास के लिए प्रक्रिया मॉडल का ज्ञान लागू करें।
2। परियोजना के लिए सॉफ्टवेयर आवश्यकता विनिर्देश (एसआरएस) दस्तावेज़ की अवधारणा को समझें
विकास।

संकाय सदस्यों के लिए दिशानिर्देश
1। शिक्षक को छात्रों को व्यावहारिक प्रदर्शन के साथ दिशानिर्देश प्रदान करना चाहिए
सभी सुविधाएं।
2। शिक्षक छात्रों को प्रयोग से संबंधित बुनियादी अवधारणाओं/सिद्धांत की व्याख्या करेगा
प्रत्येक की शुरुआतव्यावहारिक
3। प्रत्येक प्रयोग के प्रदर्शन में सभी छात्रों को शामिल करें।
4। शिक्षक से अपेक्षा की जाती है कि वे छात्रों में विकसित होने वाले कौशल और संकलन को साझा करें
और सुनिश्चित करें कि छात्रों में संबंधित कौशल और दक्षताओं को विकसित किया गया है
प्रयोग का पूरा होना।
5। शिक्षकों को छात्रों को हाथों के अनुभव के लिए अवसर देना चाहिए
प्रदर्शन।
6। शिक्षक छात्रों को अतिरिक्त ज्ञान और कौशल प्रदान कर सकता है, भले ही नहीं
ढकेलनाडी मैनुअल में लेकिन संबंधित उद्योग द्वारा छात्रों से अपेक्षित है।
7। व्यावहारिक असाइनमेंट दें और टास्क अस्सी के आधार पर छात्रों के प्रदर्शन का आकलन करें
जांचें कि यह निर्देशों के अनुसार है या नहीं।
8। शिक्षक से अपेक्षा की जाती है कि वे पाठ्यक्रम के पूर्ण पाठ्यक्रम का उल्लेख करें और दिशानिर्देशों का
पालन करें
कार्यान्वयन के लिए।

छात्रों के लिए निर्देश
1। छात्रों से अपेक्षा की जाती है कि वे संकाय द्वारा वितरित सभी सिद्धांत कक्षाओं को ध्यान से सुनें
सदस्य और COS को समझें,पाठ्यक्रम, शिक्षण और परीक्षा योजना, कौशल की सामग्री
विकसित होने के लिए सेट किया गया है।
2। छात्र समूह में काम का आयोजन करेंगे और सभी टिप्पणियों का रिकॉर्ड बनाएंगे।
3। छात्र उद्योगों द्वारा अपेक्षित पी रखरखाव कौशल को विकसित करेंगे।
4। छात्र संबंधित हाथ -कौशल विकसित करने और आत्मविश्वास का निर्माण करने का प्रयास करेगा।
5। छात्र उन लोगों के अलावा अधिक विचारों, नवाचारों, कौशल आदि को विकसित करने की आदतें विकसित
करेगा
कोमल
सॉफ्टवेयर इंजीनियरिंग

गुंजाइश में शामिल हैमैनुअल की।
6। छात्र तकनीकी पत्रिकाओं और डेटा पुस्तकों का उल्लेख करेगा।
7। छात्र को अनुसूची के अनुसार प्रयोग कार्य प्रस्तुत करने की आदत विकसित करनी चाहिए और
वह/उसे उसी के लिए अच्छी तरह से तैयार किया जाना चाहिए।

सॉफ्टवेयर इंजीनियरिंग प्रयोगशाला कार्य के लिए सामान्य दिशानिर्देश
1। छात्र को व्यावहारिक सूची में वर्णित सभी व्यावहारिक प्रदर्शन करना होगा।
2। व्यावहारिक सूची करने के लिए, छात्र व्यक्तिगत रूप से काम करने में सक्षम हो सकता है या एक टीम में काम कर सकता है
विषय शिक्षक दिशानिर्देश।3। टीम की स्थापना के बाद, हर टीम को समस्या क्षेत्र / परिभाषा की पहचान करनी होगी
प्रयोगशाला का काम करना।
4। हर टीम को 15 दिनों के भीतर संबंधित संकाय सदस्य को अपनी समस्या की परिभाषा को मंजूरी देनी होगी
सेमेस्टर की शुरुआत।
5। एक बार जब समस्या की परिभाषा Facu lty सदस्य द्वारा अनुमोदित हो जाती है, तो हर टीम को सभी प्रदर्शन करना पड़ता है
उनकी संबंधित समस्या परिभाषा के आधार पर व्यावहारिक।

कोमल
सॉफ्टवेयर इंजीनियरिंग)

अनुक्रमणिका
(प्रगतिशील मूल्यांकन पत्र)
सीनियर
प्रयोग पृष्ठ का उद्देश्य (ओं)
की तारीख
अभिनय करना
की इक्का तिथि
सब्सन करना
आयन मूल्यांकनकर्ता
टी
निशान हस्ताक्षर। का
अध्यापक
दिनांक पुनर्विवाह के साथ
केएस
1 विभिन्न प्रकार की सॉफ्टवेयर प्रक्रिया का अध्ययन
तुलना के साथ मॉडल और पता है कि कौन सा
प्रक्रिया मॉडल आपके लिए उपयुक्त होगा
चयनित परियोजना।
2 प्रोज सीटी प्रबंधन पर चर्चा करें: परियोजना
अपने बारे में योजना और परियोजना शेड्यूलिंग
परियोजना ।
3 सॉफ्टवेयर की आवश्यकता तैयार करेंजाहिर
चयनित के लिए विनिर्देश (SRS) दस्तावेज़
परियोजना।
4 अपने चयनित के लिए डेटा प्रवाह आरेख बनाएं
परियोजना।
5 अपने के लिए इकाई -Relationship आरेख बनाएं
चयनित परियोजना
6 अपने चयनित के लिए usecase आरेख ड्रा करें
परियोजना ।
7 बुनियादी लागू करके समस्या को हल करें
कोकोमो मॉडल।
8 मॉडलिंग यूएमएल वर्ग आरेख और
अनुक्रम आरेख
9। प्रदर्शन करने के लिए विभिन्न परीक्षण मामलों को डिजाइन करें
सिस्टम का परीक्षण और भी प्रदर्शन करें

विभिन्न प्रकार के टीesting।
10। किसी भी दो खुले स्रोत उपकरणों का अध्ययन
इन्फ्रास्ट्रक्चर ऑटोमेशन के लिए DevOps,
कॉन्फ़िगरेशन प्रबंधन, परिनियोजन
स्वचालन, प्रदर्शन nce प्रबंधन, लॉग
प्रबंधन की निगरानी।
कुल
कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 1
उद्देश्य: तुलना के साथ विभिन्न प्रकार के सॉफ़्टवेयर प्रक्रिया मॉडल का अध्ययन करें और पता करें कि कौन सी प्रक्रिया
मॉडल आपके चयनित परियोजना के लिए उपयुक्त होगा।
उद्देश्य: विभिन्न प्रक्रिया मॉडल सीखने और SUI की पहचान करने के लिएपरियोजना विकास के लिए तालिका मॉडल।

 सिद्धांत:
एक सॉफ्टवेयर प्रक्रिया को कार्य गतिविधियों, कार्यों और कार्यों के संग्रह के रूप में परिभाषित किया गया है जब कुछ कार्य उत्पाद बनाया जाना है, तो प्रदर्शन किया।

विभिन्न प्रक्रिया मॉडल की सूची
 झरना मॉडल।
 वी मॉडल।
 वृद्धिशील मॉडल।
 रेड मॉडल।
 एजाइल मॉडल।
 Iterative मॉडल।
 सर्पिल मॉडल।
 प्रोटोटाइप मॉडल।
 एजाइल मॉडल
प्रश्नोत्तरी:
1। झरना मॉडल और वृद्धिशील मॉडल की तुलना करें।
2। राज्य का मौसम निम्नलिखित सेंटatements सही या गलत हैं। अपने उत्तर को सही ठहराएं।
ए) सॉफ्टवेयर विकास संगठन जो के लिए पुनरावृत्ति झरना मॉडल का पालन करते हैं
उत्पाद विकास अधिकतम ग्राहकों की संतुष्टि प्रदान करता है।
बी) सर्पिल मॉडल के लिए छोरों को तय किया गया है।

सुझाए गए संदर्भ:
1। इयान सोमरविले, सॉफ्टवेयर इंजीनियरिंग, पियर्सन एजुकेशन एशिया
2। रोजर एस.प.मैन, सॉफ्टवेयर इंजीनियरिंग - एक प्रैक्टिशनर का दृष्टिकोण, मैकग्रा -हिल इंटरनेशनल संस्करणों

कोमल
सॉफ्टवेयर इंजीनियरिंग

जनमत संग्रहछात्रों द्वारा उपयोग किए जाने वाले सीईएस:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन के रूप में
पूरा पूछा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण पूरा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान पूर्ण
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
पूर्ण समझ
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
संयुक्त राष्ट्रधूर्तता

सभी को सही उत्तर
प्रश्न

संकाय के हस्ताक्षर:

कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 2
उद्देश्य: प्रोजेक टी प्रबंधन पर चर्चा करें: अपनी परियोजना के बारे में परियोजना योजना और परियोजना शेड्यूलिंग।
उद्देश्य:
1। समय के साथ परियोजना की गुंजाइश देने की योजना का प्रतिनिधित्व करने के लिए।

लिखित:

एक बार जब कोई परियोजना संभव हो जाती है, तो सॉफ्टवेयर प्रोजेक्ट मैनेजर्स प्रोजेक्ट प्लानिंग करते हैं। परियोजना
योजना बनाई जाती है और यहां तक कि पूरा किया जाता हैकिसी भी विकास गतिविधि से शुरू होता है। परियोजना की योजना बना
निम्नलिखित आवश्यक गतिविधियों के समावेश:
प्रोजेक्ट -टास्क शेड्यूलिंग एक महत्वपूर्ण प्रोजेक्ट प्लानिंग गतिविधि है। इसमें निर्णय लेना शामिल है जब कार्य तब उठाए जाएंगे। परियोजना गतिविधियों को शेड्यूल करने के लिए, एक सॉफ्टवेयर प्रोजेक्ट मैनेजर
निम्नलिखित करने की आवश्यकता है:
1। परियोजना को पूरा करने के लिए सभी कार्यों को पहचानें nee ded।
2। छोटी गतिविधियों में बड़े कार्यों को तोड़ें।
3। विभिन्न के बीच निर्भरता का निर्धारण करेंटी गतिविधियाँ।
4। गतिविधियों को पूरा करने के लिए आवश्यक समय अवधि के लिए सबसे संभावित अनुमान स्थापित करें।
5। गतिविधियों के लिए संसाधन आवंटित करें।
6। विभिन्न गतिविधियों के लिए शुरुआती और समाप्ति तिथियों की योजना बनाएं।
7। महत्वपूर्ण पथ का निर्धारण करें।

एक महत्वपूर्ण पथ गतिविधियों की श्रृंखला है जो परियोजना की अवधि निर्धारित करती है। में पहला कदम
सॉफ्टवेयर प्रोजेक्ट को शेड्यूल करने में प्रोजेक्ट को पूरा करने के लिए आवश्यक सभी कार्यों की पहचान
करना शामिल है। ए
इंट्रिकैसी का अच्छा ज्ञानप्रोजे सीटी और विकास प्रक्रिया के एस प्रबंधकों को मदद करता है
परियोजना के महत्वपूर्ण कार्यों को प्रभावी ढंग से पहचानें। अगला, बड़े कार्य एक में टूट गए हैं
छोटी गतिविधियों का तार्किक सेट जो विभिन्न इंजीनियरों को सौंपा जाएगा। काम का टूटना
संरचना औपचारिकता प्रबंधक को परियोजना प्रबंधक के बाद व्यवस्थित रूप से कार्यों को तोड़ने में मदद
करती है
कार्यों को तोड़ दिया है और कार्य टूटने की संरचना बनाई है, उसे निर्भरता खोजना होगा
गतिविधियों के बीच।
डेविभिन्न ACTI vities के बीच पेंडेंसी उस क्रम को निर्धारित करती है जिसमें विभिन्न गतिविधियाँ होंगी
किया जाएगा। यदि किसी गतिविधि ए को किसी अन्य गतिविधि बी के परिणामों की आवश्यकता होती है, तो
गतिविधि ए होनी चाहिए
गतिविधि बी के बाद अनुसूचित, सामान्य रूप से, कार्य निर्भरताएं कार्यों के बीच एक आंशिक आदेश को
परिभाषित करती हैं, अर्थात्।
प्रत्येक कार्य अन्य कार्यों के सबसेट से पहले हो सकता है, लेकिन कुछ कार्यों में कोई पूर्वता आदेश नहीं हो
सकता है
उनके बीच परिभाषित (समवर्ती कार्य कहा जाता है)। गतिविधियों के बीच निर्भरता मैंमें प्रतिनिधित्व किया
एक गतिविधि नेटवर्क का रूप। एक बार गतिविधि नेटवर्क प्रतिनिधित्व पर काम किया गया है,
प्रत्येक गतिविधि को संसाधन आवंटित किए जाते हैं।
कोमल
सॉफ्टवेयर इंजीनियरिंग

संसाधन आवंटन आमतौर पर एक गैंट चार्ट का उपयोग करके किया जाता है। संसाधन आवंटन के बाद, एक
पर्ट
चार्ट प्रतिनिधित्व विकसित किया गया है। PERT चार्ट प्रतिनिधित्व कार्यक्रम की निगरानी के लिए उपयुक्त है
और
नियंत्रण। टास्क शेड्यूलिंग के लिए, प्रोजेक्ट मैनेजर को प्रोजेक्ट कार्यों को विघटित करने की आवश्यकता
हैका एक सेट
गतिविधियाँ। जब प्रत्येक गतिविधि का प्रदर्शन किया जाना है, तो समय सीमा निर्धारित की जानी है। ईसी एच
का अंत
गतिविधि को मील का पत्थर कहा जाता है। प्रोजेक्ट मैनेजर की निगरानी करके एक परियोजना की प्रगति को
ट्रैक करता है
समय पर मील के पत्थर का पूरा होना। यदि वह देखता है कि मील के पत्थर में देरी होने लगती है, तो उसके
पास है
गतिविधियों को ध्यान से नियंत्रित करने के लिए, ताकि वह समग्र समय सीमा अभी भी पूरी हो सके।
एक गैंट चार्ट (टाइम लाइन चार्ट) एक विशेष प्रकार का बार चार्ट है जहां प्रत्येक बार का प्रतिनिधित्व करता

हैएक गतिविधि ents।
बार लाइन के साथ बार तैयार किए जाते हैं। प्रत्येक बार की लंबाई नियोजित समय की अवधि के लिए आनुपातिक है
इसी गतिविधि के लिए। सॉफ्टवेयर प्रोजेक्ट प्रबंधन में गैंट चार्ट का उपयोग वास्तव में एक है
मानक गैंट चार्ट का बढ़ाया संस्करण।

कोमल
सॉफ्टवेयर इंजीनियरिंग

प्रश्नोत्तरी:
1) प्रोजेक्ट शेड्यूलिंग प्रक्रिया की व्याख्या करें।
2) सॉफ्टवेयर लागत अनुमान के लिए उपयोग किए जाने वाले सॉफ़्टवेयर मैट्रिक्स की व्याख्या करें

सुझाए गए संदर्भ:
1। इयान सोमरविले, एस0ftware इंजीनियरिंग, पियर्सन एजुकेशन एशिया
2। रोजर एस.प.मैन, सॉफ्टवेयर इंजीनियरिंग - एक प्रैक्टिशनर का दृष्टिकोण, मैकग्रा -हिल इंटरनेशनल संस्करणों
छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन
जैसा कि पूरा पूछा गया
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण पूरा
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान पूरा

कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
विकासका
समाधान
अवधारणा स्पष्टता
और पूर्ण समझ
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता
& समझ
का सही उत्तर
सभी प्रश्न

संकाय के हस्ताक्षर:
कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 3
उद्देश्य: चयनित परियोजना के लिए सॉफ़्टवेयर आवश्यकता विनिर्देश (SRS) दस्तावेज़ तैयार करें।
उद्देश्य:
1। सीखें कि हमारे सॉफ्टवेयर उत्पाद का विस्तृत अवलोकन कैसे प्रदान करें, इसके पैरामीटर और लक्ष्य।
2। DESC रिब्स प्रोजेक्ट के लक्षित दर्शकों और इसके उपयोगकर्ता इंटरफ़ेस, हार्डवेयर और सॉफ़्टवेयर आवश्यकताओं को छोड़ देता है।
लिखित:
एक सॉफ्टवेयर आवश्यकताएँ विनिर्देश (SRS) एक दस्तावेज है जो एक विस्तृत विवरण होने पर बनाया जाता है
निर्मित किए जाने वाले सॉफ़्टवेयर के सभी पहलुओं को परियोजना शुरू करने से पहले निर्दिष्ट किया जाना चाहिए। क्या यह महत्वपूर्ण है
ई नहीं है कि एक औपचारिक एसआरएस हमेशा नहीं लिखा जाता है। वास्तव में, ऐसे कई उदाहरण हैं जिनमें प्रयास हैं
एक SRS पर खर्च किया गया बेहतर स्पेन हो सकता हैटी अन्य सॉफ्टवेयर इंजीनियरिंग गतिविधियों में।

हालांकि, जब
सॉफ्टवेयर को किसी तीसरे पक्ष द्वारा विकसित किया जाना है, जब विशिष्ट रूप से कमी की कमी से गंभीर व्यवसाय पैदा होगा
मुद्दे, या जब कोई प्रणाली बेहद जटिल या व्यवसाय महत्वपूर्ण होती है, तो एक एसआरएस उचित हो सकता है।
SRS के लिए IEEE टेम्पलेट

कोमल
सॉफ्टवेयर इंजीनियरिंग

सॉफ्टवेयर आवश्यकताएँ विनिर्देश
<प्रोजेक्ट> के लिए
संस्करण 1.0 स्वीकृत
<लेखक> द्वारा तैयार किया गया
<संगठन>
<तिथि बनाई>

एसओ एफ टी
सॉफ्टवेयर इंजीनियरिंग

विषयसूची

कोमल
सॉफ्टवेयर इंजीनियरिंग

परिचय
उद्देश्य
<उस उत्पाद की पहचान करें जिसकी सॉफ़्टवेयर आवश्यकताएं इस दस्तावेज़ में निर्दिष्ट हैं, जिसमें शामिल हैं
संशोधन या रिलीज़ नंबर। उस उत्पाद के दायरे का वर्णन करें जो इस एसआरएस द्वारा कवर किया गया है,
खासकर अगर
यह SRS सिस्टम या एक सबसिस्टम के केवल हिस्से का वर्णन करता है।>
करनाकवच
<किसी भी मानक या टाइपोग्राफिक सम्मेलनों का वर्णन करें जो इस एसआरएस को लिखते समय पालन किए
गए थे, जैसे
फोंट या हाइलाइटिंग के रूप में जिसका विशेष महत्व है। उदाहरण के लिए, बताएं कि क्या प्राथमिकताएं
अधिक हैं -
स्तर की आवश्यकताओं को विस्तृत आवश्यकताओं से विरासत में मिला है, या क्या हर आवश्यकता है
कथन की अपनी प्राथमिकता है।>

उत्पाद गुंजाइश
<सॉफ़्टवेयर का संक्षिप्त विवरण निर्दिष्ट किया जा रहा है और इसके उद्देश्य, Re levant लाभ सहितएस,
उद्देश्य और ध्येय। कॉर्पोरेट लक्ष्यों या व्यावसायिक रणनीतियों के लिए सॉफ्टवेयर से संबंधित। अगर एक
अलग दृष्टि
और स्कोप दस्तावेज़ उपलब्ध है, इसकी सामग्री को यहां डुप्लिकेट करने के बजाय इसे देखें।>
संदर्भ
<किसी भी अन्य दस्तावेज़ या वेब पते को सूचीबद्ध करें, जिसे यह SRS संदर्भित करता है। इनमें उपयोगकर्ता
शामिल हो सकता है
इंटरफ़ेस स्टाइल गाइड, अनुबंध, मानक, सिस्टम आवश्यकताएँ विनिर्देशों, केस दस्तावेजों का उपयोग करें,
या एक दृष्टि और स्कोप दस्तावेज। पर्याप्त जानकारी प्रदान करें ताकि पाठक एसीसी कर सकेएक c opy का
निबंध
शीर्षक, लेखक, संस्करण संख्या, दिनांक और स्रोत या स्थान सहित प्रत्येक संदर्भ।>
समग्र विवरण
उत्पाद परिप्रेक्ष्य
<इस SRS में निर्दिष्ट उत्पाद के संदर्भ और मूल का वर्णन करें। उदाहरण के लिए, राज्य
यह उत्पाद एक उत्पाद परिवार का एक अनुवर्ती सदस्य है, कुछ मौजूदा प्रणालियों के लिए एक प्रतिस्थापन, या
ए
नया, आत्म -उत्पादित उत्पाद। यदि SRS एक बड़ी प्रणाली के एक घटक को परिभाषित करता है, तो संबंधित
करें
मस्ती के लिए बड़ी प्रणाली की आवश्यकताएंइस सॉफ़्टवेयर की ctionality और मैं बीच के इंटरफेस को
डेंटाइज़ करता हूं
दो। एक सरल आरेख जो समग्र प्रणाली, सबसिस्टम के प्रमुख घटकों को दर्शाता है
परस्पर संबंध, और बाहरी इंटरफेस सहायक हो सकते हैं।>
उत्पाद कार्य

<प्रमुख कार्यों को सारांशित करें जो उत्पाद को प्रदर्शन करना चाहिए या म्यू सेंट उपयोगकर्ता को प्रदर्शन करना चाहिए। विवरण होगा
धारा 3 में प्रदान किया गया है, इसलिए केवल एक उच्च स्तरीय सारांश (जैसे कि बुलेट सूची) की आवश्यकता है। व्यवस्थित करना
उन्हें किसी भी पाठक के लिए समझने योग्य बनाने के लिए कार्य करता हैएसआरएस। के प्रमुख समूहों की एक तस्वीर
संबंधित अपेक्षित rements और वे कैसे संबंधित हैं, जैसे कि एक शीर्ष स्तर डेटा प्रवाह आरेख या ऑब्जेक्ट वर्ग
आरेख, अक्सर प्रभावी होता है।>
उपयोगकर्ता कक्षाएं और विशेषताएँ
<विभिन्न उपयोगकर्ता वर्गों की पहचान करें जो आप अनुमान लगाते हैं कि इस उत्पाद का उपयोग करेंगे। उपयोगकर्ता कक्षाएं हो सकती हैं
उपयोग की आवृत्ति, उपयोग किए जाने वाले उत्पाद कार्यों के सबसेट, तकनीकी विशेषज्ञता, सुरक्षा के आधार पर अंतर एड
या विशेषाधिकार स्तर, शैक्षिक स्तर, या अनुभव। प्रासंगिक परिवर्तन का वर्णन करेंप्रत्येक उपयोगकर्ता की eristics
कक्षा। कुछ आवश्यकताएं केवल कुछ उपयोगकर्ता वर्गों से संबंधित हो सकती हैं। सबसे महत्वपूर्ण भेद करें इस उत्पाद के लिए उपयोगकर्ता कक्षाएं जो संतुष्ट करने के लिए कम महत्वपूर्ण हैं।>
परिचालन लागत वातावरण
<उस वातावरण का वर्णन करें जिसमें सॉफ़्टवेयर काम करेगा, जिसमें हार्डवेयर प्लेटफॉर्म भी शामिल है, ऑपरेटिंग सिस्टम और संस्करण, और कोई अन्य सॉफ्टवेयर घटक या एप्लिकेशन जिसके साथ यह होना चाहिए
शांति से सह -अस्तित्व।>
कोमल
सॉफ्टवेयर इंजीनियरिंग

डिज़ाइनऔर कार्यान्वयन बाधाएं
<किसी भी आइटम या मुद्दों का वर्णन करें जो डेवलपर्स के लिए उपलब्ध विकल्पों को सीमित करेगा। ये हो सकते हैं
शामिल करें: कॉर्पो दर या नियामक नीतियां;
हार्डवेयर सीमाएं (समय आवश्यकताएं, स्मृति आवश्यकताएं);
अन्य अनुप्रयोगों के लिए इंटरफेस;
विशिष्ट प्रौद्योगिकियां, उपकरण और डेटाबेस का उपयोग किया जाना है; समानांतर संचालन; भाषा की आवश्यकताएं;
संचार प्रोटोकॉल; सुरक्षा विचार; डिजाइन सम्मेलनों या प्रोग्रामिंग मानकों (के लिए)
उदाहरण, अगरग्राहक का संगठन वितरित सॉफ़्टवेयर को बनाए रखने के लिए जिम्मेदार होगा)।>
उपयोगकर्ता प्रलेखन

<उपयोगकर्ता प्रलेखन घटकों (जैसे उपयोगकर्ता मैनुअल एस, ऑन -लाइन सहायता और ट्यूटोरियल) को सूचीबद्ध करें
सॉफ्टवेयर के साथ दिया जाए। किसी भी ज्ञात उपयोगकर्ता प्रलेखन वितरण प्रारूपों की पहचान करें या मानक।>
मान्यताओं और निर्भरता
<किसी भी ग्रहण किए गए कारकों को सूचीबद्ध करें (ज्ञात तथ्यों के विपरीत) जो कि आर समानता को प्रभावित कर सकते हैं
Srs। इनमें तीसरा -pa शामिल हो सकता हैrty या वाणिज्यिक घटक जो आप उपयोग करने की योजना बनाते हैं, आसपास के मुद्दे
विकास या परिचालन वातावरण, या बाधाएं। परियोजना को प्रभावित किया जा सकता है अगर ये धारणाएँ गलत हैं, साझा नहीं की जाती हैं, या परिवर्तन करते हैं। परियोजना के पास किसी भी निर्भरता की पहचान भी करें
बाहरी कारकों पर, जैसे कि सॉफ्टवेयर घटक जिन्हें आप किसी अन्य परियोजना से पुन: उपयोग करने का इरादा रखते हैं, जब तक
वे पहले से ही कहीं और प्रलेखित हैं (उदाहरण के लिए, दृष्टि और स्कोप दस्तावेज़ या परियोजना में योजना)।>
बाह्य इंटरफैकई आवश्यकताएँ
उपयोगकर्ता इंटरफेस
<सॉफ्टवेयर उत्पाद और उपयोगकर्ताओं के बीच प्रत्येक इंटरफ़ेस की तार्किक विशेषताओं का वर्णन करें। इसमें सैंपल स्क्रीन इमेज, कोई भी जीयूआई मानक या उत्पाद परिवार शैली के गाइड शामिल हो सकते हैं पालन किया जाए, स्क्रीन लेआउट बाधाएं, मानक बटन और फ़ंक्शंस (जैसे, सहायता) जो दिखाई देंगे प्रत्येक स्क्रीन, कीबोर्ड शॉर्टकट, त्रुटि संदेश प्रदर्शन मानकों, और इसी तरह। सॉफ्टवेयर को परिभाषित करें ऐसे घटक जिसके लिए एक उपयोगकर्ता इंटरफ़ेस की आवश्यकता होती है। विवरण ओf उपयोगकर्ता इंटरफ़ेस डिजाइन होना चाहिए
एक अलग उपयोगकर्ता इंटरफ़ेस विनिर्देश में प्रलेखित।>
हार्डवेयर इंटरफेस
<सॉफ्टवेयर उत्पाद के बीच प्रत्येक इंटरफ़ेस की तार्किक और भौतिक विशेषताओं का वर्णन करें और सिस्टम के हार्डवेयर घटक। इसमें समर्थित डिवाइस प्रकार, की प्रकृति शामिल हो सकती है
सॉफ्टवेयर और हार्डवेयर, और संचार प्रोटोकॉल के बीच डेटा और नियंत्रण इंटरैक्शन
इस्तेमाल किया जाना है।>
सॉफ्टवेयर इंटरफेस
<इसके बीच संबंधों का वर्णन करेंउत्पाद और अन्य विशिष्ट सॉफ्टवेयर घटक (नाम और संस्करण), डेटाबेस, ऑपरेटिंग सिस्टम, टूल, लाइब्रेरी और एकीकृत वाणिज्यिक सहित
अवयव। सिस्टम में आने वाले डेटा आइटम या संदेशों को पहचानें और बाहर जाकर वर्णन करें
प्रत्येक का उद्देश्य। आवश्यक सेवाओं और संचार की प्रकृति का वर्णन करें। को देखें
दस्तावेज़ जो विस्तृत एप्लिकेशन प्रोग्रामिंग इंटरफ़ेस प्रोटोकॉल का वर्णन करते हैं। डेटा की पहचान करें जो होगा
सॉफ्टवेयर घटकों में साझा किया गया। अगर डेटा एसहारिंग तंत्र को एक विशिष्ट में लागू किया जाना चाहिए

तरीका (उदाहरण के लिए, एक मल्टीटास्किंग ऑपरेटिंग सिस्टम में एक वैश्विक डेटा क्षेत्र का उपयोग), इसे एक के रूप में निर्दिष्ट करें
कार्यान्वयन बाधा।>
संचार इंटरफेस
<इस उत्पाद द्वारा आवश्यक किसी भी संचार कार्यों से जुड़ी आवश्यकताओं का वर्णन करें,
ई -मेल, वेब ब्राउज़र, नेटवर्क सर्वर संचार प्रोटोकॉल, इलेक्ट्रॉनिक रूप, और इसी तरह सहित।
किसी भी प्रासंगिक संदेश स्वरूपण को परिभाषित करें। किसी भी संचार मानकों की पहचान करें जो विल।
का उपयोग किया जाना चाहिए, ऐसा
कोमल
सॉफ्टवेयर इंजीनियरिंग

एफ़टीपी या http के रूप में। किसी भी COM COMICATION सुरक्षा या एन्क्रिप्शन मुद्दों, डेटा ट्रांसफर दरों और को निर्दिष्ट करें
सिंक्रनाइज़ेशन मैकेनिज्म।>
प्रणाली की सुविधाएँ
<यह टेम्पलेट सिस्टम सुविधाओं द्वारा उत्पाद के लिए कार्यात्मक आवश्यकताओं को व्यवस्थित करने के लिए दिखाता है,
उत्पाद द्वारा प्रदान की गई प्रमुख सेवाएं। आप उपयोग के मामले, मोड द्वारा इस अनुभाग को व्यवस्थित करना पसंद कर सकते हैं
ऑपरेशन, उपयोगकर्ता वर्ग, ऑब्जेक्ट क्लास, कार्यात्मक पदानुक्रम, या संयोजनएसई, जो भी बनाता है आपके उत्पाद के लिए सबसे तार्किक अर्थ।>
सिस्टम फीचर 1
<वास्तव में यह मत कहो कि "सिस्टम एफ ईट्योर 1." कुछ शब्दों में फीचर नाम बताएं।>
4.1.1 विवरण और प्राथमिकता
<सुविधा का संक्षिप्त विवरण प्रदान करें और इंगित करें कि क्या यह उच्च, मध्यम या निम्न है
प्राथमिकता। आप विशिष्ट प्राथमिकता घटक रेटिंग भी शामिल कर सकते हैं, जैसे कि लाभ,
जुर्माना, लागत, और जोखिम (प्रत्येक के सापेक्ष पैमाने पर 1 के निचले स्तर से 9 के उच्च स्तर पर रेटेड)>>>>>
4.1.2 उत्तेजना/सम्मानआज़ाद अनुक्रम
<उपयोगकर्ता कार्यों और सिस्टम प्रतिक्रियाओं के अनुक्रमों को सूचीबद्ध करें जो परिभाषित व्यवहार को उत्तेजित करते हैं
इस सुविधा के लिए। Th ese w बीमार उपयोग के साथ जुड़े संवाद तत्वों के अनुरूप हैं
मामले।>
4.1.3 कार्यात्मक आवश्यकताएं
<इस सुविधा से जुड़ी विस्तृत कार्यात्मक आवश्यकताओं को आइटम करें। ये हैं
सॉफ्टवेयर क्षमताएं जो उपयोगकर्ता को सेवाओं को बाहर करने के लिए मौजूद होनी चाहिए
सुविधा द्वारा प्रदान किया गया, या उपयोग के मामले को निष्पादित करने के लिए। उत्पाद कैसे शामिल

करेंचाहिए
प्रत्याशित त्रुटि शर्तों या अमान्य इनपुटों का जवाब दें। आवश्यकताएं संक्षिप्त होनी चाहिए,
पूर्ण, अस्पष्ट, सत्यापन योग्य और आवश्यक। एक प्लेसहोल्डर के रूप में "टीबीडी" का उपयोग करें
संकेत दें कि जब आवश्यक जानकारी अभी उपलब्ध नहीं है।>

<प्रत्येक आवश्यकता को एक अनुक्रम संख्या या एक सार्थक टैग के साथ विशिष्ट रूप से पहचाना जाना चाहिए
किसी तरह।>

Req -1:
Req -2:
सिस्टम फ़ीचर 2 (और इसी तरह)
अन्य nonfunctional आवश्यकताएँ irements
प्रदर्शन आवश्यकताएँ
<यदि प्रदर्शन की आवश्यकता हैविभिन्न परिस्थितियों में उत्पाद के लिए irements, उन्हें यहां बताएं
और उनके तर्क को समझाएं, डेवलपर्स को इरादे को समझने और उपयुक्त डिजाइन बनाने में मदद करने के लिए
विकल्प। वास्तविक समय प्रणालियों के लिए टीआई मिंग संबंधों को निर्दिष्ट करें। इस तरह की आवश्यकताओं को विशिष्ट के रूप में बनाएं
संभव। आपको व्यक्तिगत कार्यात्मक आवश्यकताओं के लिए प्रदर्शन आवश्यकताओं को राज्य करने की आवश्यकता हो सकती है या
सुविधाएँ।>
सुरक्षा आवश्यकताओं
<उन आवश्यकताओं को निर्दिष्ट करें जो संभावित नुकसान, क्षति या नुकसान से संबंधित हैंपरिणाम हो सकता है
उत्पाद के उपयोग से। किसी भी सुरक्षा उपायों या कार्यों को परिभाषित करें, जिसे लिया जाना चाहिए, साथ ही साथ कार्रवाई भी करें
इसे रोका जाना चाहिए। किसी भी बाहरी नीतियों या नियमों को देखें जो राज्य सुरक्षा मुद्दों को प्रभावित करते हैं
उत्पाद का डिज़ाइन या उपयोग। किसी भी सुरक्षा प्रमाणपत्र को परिभाषित करें जो संतुष्ट होना चाहिए।>
कोमल
सॉफ्टवेयर इंजीनियरिंग

सुरक्षा आवश्यकताएँ
<उत्पाद के उपयोग के आसपास की सुरक्षा या गोपनीयता के मुद्दों के बारे में कोई भी आवश्यकताएं निर्दिष्ट करें या
दा का संरक्षणTA का इस्तेमाल किया या टी वह उत्पाद द्वारा बनाया गया। किसी भी उपयोगकर्ता पहचान प्रमाणीकरण को परिभाषित करें
आवश्यकताएं। किसी भी बाहरी नीतियों या नियमों को प्रभावित करने वाले नियमों का संदर्भ लें जो प्रभावित

करते हैं
उत्पाद। किसी भी सुरक्षा या गोपनीयता प्रमाणपत्रों को परिभाषित करें जो संतुष्ट होना चाहिए।>
सॉफ्टवेयर गुणवत्ता अटार ibutes
<उत्पाद के लिए किसी भी अतिरिक्त गुणवत्ता विशेषताओं को निर्दिष्ट करें जो या तो महत्वपूर्ण होगा
ग्राहक या डेवलपर्स। कुछ पर विचार करने के लिए हैं: अनुकूलनशीलता, उपलब्धता, शुद्धता,
लचीलापन,अंतर, स्थिरता, पोर्टेबिलिटी, विश्वसनीयता, पुन: प्रयोज्यता, मजबूती, परीक्षण और प्रयोज्य।
जब संभव हो तो इन्हें विशिष्ट, मात्रात्मक और सत्यापित होने के लिए लिखें। कम से कम, रिश्तेदार को स्पष्ट करें
विभिन्न विशेषताओं के लिए प्राथमिकताएं, जैसे कि एल कमाई में आसानी से उपयोग में आसानी।>
व्यावसायिक नियम
<उत्पाद के बारे में किसी भी ऑपरेटिंग सिद्धांतों को सूचीबद्ध करें, जैसे कि व्यक्ति या भूमिकाएँ जो प्रदर्शन कर सकते हैं
विशिष्ट परिस्थितियों में कार्य। ये कार्यात्मक आवश्यकता नहीं हैंअपने आप में rements, लेकिन वे
नियमों को लागू करने के लिए कुछ फ़ंक्शन अल आवश्यकताओं का अर्थ हो सकता है।>
अन्य आवश्यकताएँ
<SRS में कहीं और कवर नहीं की गई किसी भी अन्य आवश्यकताओं को परिभाषित करें। इसमें डेटाबेस शामिल हो सकता है
आवश्यकताओं, अंतर्राष्ट्रीयकरण आवश्यकताओं, कानूनी आवश्यकताओं, परियोजना के लिए उद्देश्यों का पुन: उपयोग,
और इसी तरह। कोई भी नया खंड जोड़ें जो परियोजना के लिए प्रासंगिक हैं।>
परिशिष्ट A: शब्दावली
<एसआरएस की ठीक से व्याख्या करने के लिए आवश्यक सभी शर्तों को परिभाषित करें, जिसमें समरूप और ए शामिल हैंbbreviations।
आप एक अलग शब्दावली का निर्माण करना चाह सकते हैं जो कई प्रोजेक टीएस या पूरे संगठन को फैलाता है, और
बस प्रत्येक SRS में एक एकल परियोजना के लिए विशिष्ट शब्द शामिल करें।>
परिशिष्ट बी: विश्लेषण मॉडल
<वैकल्पिक रूप से, किसी भी प्रासंगिक विश्लेषण मॉडल को शामिल करें, जैसे डेटा प्रवाह आरेख, वर्ग आरेख, राज्य -
संक्रमण आरेख, या इकाई -Relationship आरेख।>
परिशिष्ट C: निर्धारित सूची के लिए
<TBD की एक संख्या की सूची एकत्र करें (निर्धारित किए जाने के लिए) संदर्भ जो SRS में रहते हैंतो वे कर सकते हैं
बंद करने के लिए ट्रैक किया जाए।>

कोमल
सॉफ्टवेयर इंजीनियरिंग

प्रश्नोत्तरी:
1। अच्छे एसआरएस के गुण कौन से हैं?
2। कार्यात्मक एक nd गैर -फ़ंक्शनल आवश्यकता क्या है?

सुझाए गए संदर्भ:
1। इयान सोमरविले, सॉफ्टवेयर इंजीनियरिंग, पियर्सन एजुकेशन एशिया
2। रोजर एस.प.मैन, सॉफ्टवेयर इंजीनियरिंग - एक प्रैक्टिशनर का दृष्टिकोण, मैकग्रा -हिल इंटरनेशनल संस्करणों

छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
एमarks पूरा
कार्यान्वयन
जैसा कि पूरा पूछा गया
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण पूरा
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान पूरा
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता
और पूर्ण समझ
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण

का विकास
समाधान
अवधारणा स्पष्टता
& समझ
का सही उत्तर
सभी प्रश्न

संकाय के हस्ताक्षर:

कोमल
इसलिएफीटवेयर इंजीनियरिंग (3161605)

व्यावहारिक - 4
उद्देश्य: अपने चयनित परियोजना के लिए डेटा प्रवाह आरेख बनाएं।
 उद्देश्य:
डेटा प्रवाह आरेखों के माध्यम से प्रवाह उन्मुख मॉडल सीखने के लिए।

 सिद्धांत:
DFD एक सिस्टम का एक इनपुट -प्रोसेस -आउटपुट दृश्य लेता है। अर्थात्, डेटा ऑब्जेक्ट्स में बहते हैं सॉफ्टवेयर, प्रसंस्करण तत्वों द्वारा रूपांतरित होते हैं, और परिणामी डेटा ऑब्जेक्ट्स से बाहर प्रवाहित होते हैं सॉफ़्टवेयर। डेटा प्रवाह आरेख आपको सूचना डोमेन के मॉडल विकसित करने में सक्षम बनाता है और कार्यात्मक डीओमैन।
शब्द -संकेतन टिप्पणी
बाहरी
इकाई
बाहरी इकाई का नाम आयत के अंदर लिखा गया है
प्रक्रिया
प्रक्रिया का नाम सर्कल के अंदर लिखा गया है
डेटा भंडारण
एक बाईं -खुली हुई आयत को डेटा स्टोर के रूप में निरूपित किया जाता है; डेटा का नाम
स्टोर आकार के अंदर लिखा गया है
डेटा प्रवाह
डेटा प्रवाह को अपने डेटा नाम के साथ एक निर्देशित चाप द्वारा दर्शाया जाता है

कोमल
सॉफ्टवेयर इंजीनियरिंग

DFD में प्रयुक्त प्रतीकों की व्याख्या
 प्रक्रिया: प्रक्रियाएँ represe हैंसर्कल द्वारा nted। प्रक्रिया का नाम सर्कल में लिखा गया है।
प्रक्रिया का नाम आमतौर पर इस तरह से दिया जाता है जो की कार्यक्षमता का प्रतिनिधित्व करता है
प्रक्रिया। यदि आवश्यक हो तो अधिक विस्तृत कार्यक्षमता अगले स्तर में दिखाया जा सकता है। आमतौर पर
यह
प्रक्रियाओं की संख्या 7 से कम रखने के लिए बेहतर है। यदि हम देखते हैं कि प्रक्रियाओं की संख्या
7 से अधिक हो जाता है तो हमें कुछ प्रक्रियाओं को कम करने के लिए एक एकल को संयोजित करना चाहिए
प्रक्रियाओं की संख्या और आगे विघटित ईअगले स्तर तक।
 बाहरी इकाई: बाहरी संस्थाएं केवल संदर्भ आरेख में दिखाई देती हैं। बाहरी संस्थाएं हैं
एक आयत और बाहरी इकाई का नाम आकार में लिखा गया है। इन
संसाधित होने के लिए डेटा भेजें और फिर से संसाधित डेटा प्राप्त करें।
 डेटा स्टोर: डेटा स्टार्स को एक बाएं -खुले आयत द्वारा दर्शाया जाता है। डेटा स्टोर का नाम है
खुली आयत की दो क्षैतिज रेखाओं के बीच में लिखा गया। डेटा स्टोर का उपयोग रिपॉजिटरी के रूप में किया
जाता है
किस से डीएटीए को एक प्रक्रिया से या बाहर या बाहर उड़ाया जा सकता है।
 डेटा प्रवाह: डेटा प्रवाह को डेटा प्रवाह के दो घटकों के बीच एक निर्देशित किनारे के रूप में दिखाया गया है
आरेख। डेटा बाहरी इकाई से प्रक्रिया, डेटा स्टोर से प्रक्रिया से प्रक्रिया तक, दो के बीच में प्रवाहित हो सकता
है
प्रक्रियाएं और इसके विपरीत।
।
 पृष्ठभूमि / तैयारी:
DFD का स्तर
DFD पारदर्शिता बनाए रखने के लिए पदानुक्रम का उपयोग करता है इस प्रकार बहुस्तरीय DFD का
बनाया जा सकता है। DFD का स्तर
निम्नानुसार हैं:
 0-स्तरीय DFD: प्राथमिक बाहरी संस्थाएं (Boxes) सिस्टम द्वारा उपयोग के लिए जानकारी का उत्पादन
करें
और सिस्टम द्वारा उत्पन्न जानकारी का उपभोग करें
 1-स्तरीय DFD: यह सिस्टम के मुख्य कार्यों का प्रतिनिधित्व करता है और वे प्रत्येक के साथ कैसे बातचीत
करते हैं
अन्य।
 2-स्तरीय DFD: यह सिस्टम के प्रत्येक फ़ंक्शन के भीतर प्रक्रियाओं का प्रतिनिधित्व करता है और वे कैसे
बातचीत करते हैं
एक दूसरे के साथ।

 टूल एस / सामग्री की आवश्यकता:
ओ हार्डवेयर:
ओ सॉफ्टवेयर:

कोमल
सॉफ्टवेयर इंजीनियरिंग

प्रश्नोत्तरी:

1। एक डेटा प्रवाह आरेख में, एक तीर एक का प्रतिनिधित्व करता हैनियंत्रण का प्रवाह या कुछ और?
2। "सूचना प्रवाह निरंतरता" क्या है और इसे डेटा प्रवाह आरेख के रूप में कैसे लागू किया जाता है परिष्कृत?
3। DFD के क्या फायदे हैं?

सुझाए गए संदर्भ:
1। रोजर एस प्रेसमैन, सॉफ्टवेयर इंजीनियरिंग - एक व्यवसायी का दृष्टिकोण, मैकग्रा -हिल अंतर्राष्ट्रीय संस्करण
2। राजिब मॉल, सॉफ्टवेयर इंजीनियरिंग के बुनियादी बातें, भारत का प्रेंटिस हॉल।
छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशानपूरा
कार्यान्वयन के रूप में
पूरा पूछा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण पूरा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान पूर्ण
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान

अवधारणा स्पष्टता और
पूर्ण समझ
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
समझ
का सही उत्तर
सभी प्रश्न

संकाय का हस्ताक्षर
कोमल
सॉफ्टवारी इंजीनियरिंग (3161605)

व्यावहारिक - 5
उद्देश्य: अपने चयनित परियोजना के लिए इकाई -relationship आरेख बनाएं।
 उद्देश्य:
1। इकाई सेट, उनकी विशेषताओं और विभिन्न रिश्तों की पहचान करें
2। ईआर आरेख के माध्यम से डेटा मॉडल का प्रतिनिधित्व करें

 सिद्धांत:
Entity -Relationship मॉडल का उपयोग डेटाबेस के तार्किक डिजाइन का प्रतिनिधित्व करने के लिए किया जाता है। एर में
मॉडल, वास्तविक दुनिया की वस्तुएं (या अवधारणाएं) संस्थाओं के रूप में अमूर्त हैं, और अलग -अलग संभव उनमें से एसोसिएशन मोड हैंरिश्तों के रूप में नेतृत्व किया।

उदाहरण के लिए, छात्र और एससी हूल - वे दो संस्थाएं हैं। छात्र स्कूल में अध्ययन करते हैं। तो, ये दोनों संस्थाएं एक संबंध "अध्ययन" से जुड़ी हैं।

एक अन्य उदाहरण के रूप में, एक ऐसी प्रणाली पर विचार करें जहां हर रात कुछ नौकरी चलती है, जो अपडेट करती है
डेटाबेस। यहां, नौकरी और डेटाबेस दो संस्थाएं हो सकती हैं। वे रिश्ते से जुड़े हैं
"अपडेट"।

इकाई सेट और संबंध सेट

एक इकाई सेट सभी समान संस्थाओं का एक संग्रह है। के लिएउदाहरण, "छात्र" एक इकाई सेट है सभी छात्रों को सार। राम, जॉन इस सेट से संबंधित विशिष्ट संस्थाएं हैं। इसी तरह, ए "रिलेशनशिप" सेट समान रिश्तों का एक सेट है।

संस्था की विशेषताएँ

विशेषताएँ किसी इकाई सेट से संबंधित किसी भी इकाई का वर्णन करने वाली विशेषताएं हैं। एक सेट में कोई भी इकाई z ero या अधिक विशेषताओं द्वारा वर्णित किया जा सकता है।

उदाहरण के लिए, किसी भी छात्र को एक नाम, उम्र, एक पता मिला है। किसी भी समय एक छात्र अध्ययन कर सकता है केवल एक स्कूल में। स्कूल मेंउसके पास एक रोल नंबर होगा, और निश्चित रूप से एक ग्रेड जिसमें वह अध्ययन करते हैं। ये डेटा इकाई सेट छात्र की विशेषताएं हैं।

चाबी

किसी इकाई सेट की एक या अधिक विशेषता का उपयोग निम्न कुंजियों को परिभाषित करने के लिए किया जा सकता है:

सुपर कुंजी: एक या एक से अधिक विशेषताएं, जो एक साथ होने पर, एक इकाई की पहचान करने में मदद करती हैं एक इकाई सेट में। उदाहरण के लिए, एक स्कूल में किसी भी संख्या में छात्र हो सकते हैं। हालांकि, अगर हम जानते हैं ग्रेड और रोल नंबर, फिर हम विशिष्ट रूप से उस SCH में एक छात्र की पहचान कर सकते हैंऊल।
कोमल
सॉफ्टवेयर इंजीनियरिंग

उम्मीदवार कुंजी: यह एक सुपर कुंजी का न्यूनतम सबसेट है। दूसरे शब्दों में, एक सुपर कुंजी में हो सकता है एक्सट्रॉनियस विशेषताओं, व्हिक एच किसी वस्तु को विशिष्ट रूप से पहचानने में मदद नहीं करते हैं। जब ऐसी विशेषताएँ हटाए गए हैं, इसलिए की गई कुंजी को एक उम्मीदवार कुंजी कहा जाता है।

प्राथमिक कुंजी: एक डेटाबेस में एक से अधिक उम्मीदवार कुंजी हो सकती है। किसी भी उम्मीदवार कुंजी के लिए चुना गया डेटाबेस के विशेष कार्यान्वयन को प्राथमिक कुंजी कहा जाता है।

प्राइम विशेषता: कोई भी अट्रीब्यूट एक सुपर कुंजी में भाग ले रहा है

कमजोर इकाई

एक इकाई सेट को कमजोर कहा जाता है यदि यह किसी अन्य इकाई सेट पर निर्भर है। एक कमजोर इकाई नहीं हो सकती विशिष्ट रूप से केवल यह विशेषताओं द्वारा पहचाना जाता है। दूसरे शब्दों में, इसमें एक सुपर कुंजी नहीं है।

उदाहरण के लिए, एक ऐसी कंपनी पर विचार करें जो कर्मचारियों को उनके लिए यात्रा भत्ता देने की अनुमति देती है
सगा परिवार। तो, यहां हमारे पास दो इकाई सेट हैं: कर्मचारी और परिवार, "दावा कर सकते हैं
के लिए "। हालांकि, परिवार के पास एक सुपर कुंजी नहीं है। एक परिवार का अस्तित्वपूरी तरह से निर्भर है
संबंधित कर्मचारी। तो, यह केवल कर्मचारी के संदर्भ में सार्थक है।

इकाई सामान्यीकरण और विशेषज्ञता
एक बार जब हमने इकाई सेटों की पहचान कर ली, तो हम उनमें से कुछ समानता पा सकते हैं। उदाहरण के लिए,
कई व्यक्ति एक बैंकिंग प्रणाली के साथ बातचीत करता है। उनमें से अधिकांश ग्राहक हैं, और बाकी कर्मचारी या
अन्य सेवा प्रदाता। यहां, ग्राहक, कर्मचारी व्यक्ति हैं, लेकिन कुछ विशेषज्ञता के साथ।
या अन्य तरीके से, व्यक्ति सामान्यीकृत रूप हैग्राहक और कर्मचारी इकाई सेट।

ईआर मॉडल विशेषज्ञता (और इस प्रकार, सामान्यीकरण) को चित्रित करने के लिए "आईएसए" पदानुक्रम का उपयोग करता है।

मानचित्रण कार्डिनलिटीज
ईआर मॉडलिंग के मुख्य कार्यों में से एक विभिन्न इकाई सेटों को जोड़ना है। आइए दो इकाई पर विचार करें
E1 और E2 में संस्थाओं की संख्या के आधार पर एक संबंध सेट R द्वारा जुड़े E1 और E2 सेट हैं।
साथ जुड़े, हमारे पास निम्नलिखित चार प्रकार के मैपिंग हो सकते हैं:

एक से एक: E1 में एक इकाई सबसे अधिक एक इकाई से संबंधित हैE2, और इसके विपरीत
एक से कई: E1 में एक इकाई E2 में शून्य या अधिक संस्थाओं से संबंधित हो सकती है। E2 में कोई भी इकाई हो सकती है
E1 में अधिकांश एकल इकाई से संबंधित हो।
कई से एक: E1 में शून्य या अधिक संख्या में संस्थाएं E2 में एक ही इकाई से जुड़ी हो सकती हैं।
हालांकि, E2 में एक इकाई E1 में अधिकांश एक इकाई से संबंधित हो सकती है।
बहुत से कई: किसी भी संख्या में संस्थाएं E2 में किसी भी संख्या में संस्थाओं से संबंधित हो सकती हैं, जिसमें शामिल हैं
शून्य, और इसके विपरीत।
ईआर आरेख
किसी दिए गए समस्या से सेंटहम p ossible इकाई सेट, उनकी विशेषताओं और की पहचान करते हैं
विभिन्न इकाई सेटों के बीच संबंध। एक बार जब हमारे पास ये जानकारी होती है, तो हम उनका प्रतिनिधित्व करते हैं
चित्रात्मक रूप से, एक इकाई -Relationship (ER) आरेख कहा जाता है।
कोमल
सॉफ्टवेयर इंजीनियरिंग

प्रश्नोत्तरी:
1। कमजोर एंटिटीसेट क्या है?
2। मैपिंग कार्डिनल इट क्या है?

छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

कोमल
सॉफ्टवेयर इंजीनियरिंग

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान COMPलेट
कार्यान्वयन
जैसा कि पूरा पूछा गया
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण पूरा
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान पूरा
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता
और पूर्ण समझ
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण

का विकास
समाधान
अवधारणा स्पष्टता
& समझ
का सही उत्तर
सभी प्रश्न

संकाय के हस्ताक्षर:

इसलिएफुट
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 6
उद्देश्य: अपने चयनित परियोजना के लिए USECASE आरेख बनाएं।
 उद्देश्य:
1। सिस्टम के निष्पादन के विभिन्न परिदृश्यों को लिखने के लिए।
2। विभिन्न UML का पता लगाने के लिए USECASE आरेख को आकर्षित करने के लिए केस आरेख घटकों का उपयोग करें।

 सिद्धांत:
o किसी सिस्टम के गतिशील व्यवहार का प्रतिनिधित्व करने के लिए एक उपयोग केस आरेख का उपयोग किया जाता है। यह एनकैप्सुलेट करता है
उपयोग के मामलों, अभिनेताओं और उनके संबंधों को शामिल करके सिस्टम की कार्यक्षमता। यह कार्यों, सेवाओं को मॉडल करें,और एक s ystem/सबसिस्टम द्वारा आवश्यक कार्य
आवेदन पत्र। यह एक प्रणाली की उच्च -स्तरीय कार्यक्षमता को दर्शाता है और यह भी बताता है कि उपयोगकर्ता कैसे
एक प्रणाली को संभालता है।
o उपयोग केस आरेख का उद्देश्य
 एक उपयोग केस आरेख का मुख्य उद्देश्य एक के गतिशील पहलू को चित्रित करना है
प्रणाली। यह संचय प्रणाली की आवश्यकता को पूरा करता है, जिसमें आंतरिक दोनों शामिल हैं
साथ ही बाहरी प्रभाव। यह व्यक्तियों को आमंत्रित करता है, मामलों का उपयोग करता है, और कई चीजें

अभिनेताओं और तत्वों के लिए जवाबदेहउपयोग के मामले का कार्यान्वयन
आरेख। यह दर्शाता है कि बाहरी वातावरण से एक इकाई कैसे बातचीत कर सकती है
सिस्टम के एक हिस्से के साथ।
निम्नलिखित एक उपयोग केस आरेख के उद्देश्य नीचे दिए गए हैं:
1। यह सिस्टम की जरूरतों को इकट्ठा करता है।
2। यह सिस्टम के बाहरी दृश्य को दर्शाता है।
3। यह आंतरिक और साथ ही बाहरी कारकों को पहचानता है जो सिस्टम को प्रभावित करते हैं।
4। यह अभिनेताओं के बीच बातचीत का प्रतिनिधित्व करता है।
o एक उपयोग -कैस आरेख में, एक अभिनेता सिस्टम का एक उपयोगकर्ता है (यानी SOMके लिए बाहरी
प्रणाली; एक विशेष भूमिका में मानव या गैर -मानव) अभिनय हो सकता है।
o एक उपयोग-केस एक ऐसा कार्य है जिसे अभिनेता को सिस्टम की मदद से प्रदर्शन करने की आवश्यकता
है,
जैसे, किसी पुस्तक का विवरण खोजें या एक बुकशॉप में रसीद की एक प्रति प्रिंट करें।
o हम सिस्टम को निरूपित करने के लिए उपयोग के मामलों के एक सेट के आसपास एक बॉक्स (एक लेबल
के साथ) खींच सकते हैं
सीमा, जैसा कि पिछले एस लाइड ("लाइब्रेरी सिस्टम") पर है।

अभिनेताओं के बीच विरासत का उपयोग यह दिखाने के लिए किया जा सकता है कि एक अभिनेता के सभी
उपयोग के मामले हैं
ओटी के लिए उपलब्ध हैउसकी:
यदि कई उपयोग के मामलों में शामिल हैं, तो उनकी कार्यक्षमता के हिस्से के रूप में, एक और उपयोग का
मामला, हमारे पास ए
यह एक उपयोग के साथ दिखाने के लिए विशेष तरीका है -एक << शामिल >> संबंध के साथ।
कोमल
सॉफ्टवेयर इंजीनियरिंग

यदि एक उपयोग -कैस के दो या अधिक अलग -अलग परिणाम हैं, तो हम इसे दिखा सकते हैं
एक मुख्य उपयोग मामले और एक या अधिक सहायक मामलों के लिए उपयोग के मामले का विस्तार करना।

 पृष्ठभूमि / तैयारी:
कैसे एक उपयोग केस आरेख आकर्षित करने के लिए?
पूरे सिस्टम बी का विश्लेषण करना आवश्यक हैएक उपयोग केस आरेख को आकर्षित करने के साथ शुरू
करें, और
तब सिस्टम की कार्यक्षमता पाई जाती है। और एक बार हर एक कार्यक्षमता की पहचान हो जाती है, वे
फिर उपयोग केस आरेख में उपयोग किए जाने वाले मामलों में उपयोग किए जाते हैं।
उसके बाद, हम उन अभिनेताओं को सूचीबद्ध करेंगे जो सिस्टम के साथ बातचीत करेंगे। अभिनेता व्यक्ति हैं
या

एक ऐसी चीज जो एक सिस्टम की कार्यक्षमता को आमंत्रित करती है। यह एक प्रणाली या एक निजी इकाई हो सकती है, जैसे कि
इसके लिए एक इकाई की आवश्यकता होती है जो कार्यात्मक के लिए प्रासंगिक होउस प्रणाली के लिए जिस पर वह जा रहा है
इंटरैक्ट करना।
एक बार अभिनेताओं और उपयोग के मामलों को सूचीबद्ध करने के बाद, अभिनेता और उपयोग केस के बीच संबंध/
सिस्टम का निरीक्षण किया जाता है। यह उन समय की पहचान करता है जो एक अभिनेता सिस्टम के साथ संवाद करता है।
मूल रूप से, एक अभिनेता किसी विशेष उदाहरण पर उपयोग के मामले या सिस्टम के साथ कई बार बातचीत कर सकता है
समय का।
निम्नलिखित कुछ नियम हैं जिनका उपयोग एक उपयोग केस आरेख को चित्रित करते समय किया जाना चाहिए:
1। एक प्रासंगिक और सार्थक नाम को एक्टो को सौंपा जाना चाहिएआर या एक का उपयोग मामला प्रणाली।
2। एक उपयोग मामले के साथ एक अभिनेता के संचार को एक समझ में परिभाषित किया जाना चाहिए रास्ता।
3। आवश्यक होने पर और जब आवश्यक हो तो निर्दिष्ट नोटेशन।
4। सबसे महत्वपूर्ण बातचीत को कई के बीच प्रतिनिधित्व किया जाना चाहिए
उपयोग के मामले और अभिनेताओं के बीच बातचीत।
उपयोग केस आरेखों के उद्देश्य इस प्रकार हो सकते हैं:

 एक प्रणाली की आवश्यकताओं को इकट्ठा करने के लिए उपयोग किया जाता है।
 एक प्रणाली के बाहरी दृश्य को प्राप्त करने के लिए उपयोग किया जाता है।
 बाहरी और आंतरिक एफए की पहचान करेंसिस्टम को प्रभावित करने वाले ctors।
 शो टी वह आवश्यकताओं के बीच बातचीत करना अभिनेता हैं।
परिदृश्यों
• परिदृश्य वास्तविक -जीवन के उदाहरण हैं कि एक प्रणाली का उपयोग कैसे किया जा सकता है।
• उन्हें शामिल करना चाहिए
- शुरुआती स्थिति का विवरण;
- घटनाओं के सामान्य प्रवाह का विवरण;
- क्या गलत हो सकता है इसका विवरण;
- अन्य समवर्ती गतिविधियों के बारे में जानकारी;
कोमल
सॉफ्टवेयर इंजीनियरिंग

जब परिदृश्य खत्म हो जाता है तो राज्य का विवरण।
 उपकरण / सामग्रीआवश्यकता है:
ओ हार्डवेयर:
ओ सॉफ्टवेयर:

 प्रक्रिया / चरण:
o उपयोग के मामले विकसित करना:
o चरण एक - कहानी में शामिल अभिनेताओं के सेट को परिभाषित करें
 अभिनेता लोग, उपकरण, या अन्य सिस्टम हैं जो सिस्टम या उत्पाद का उपयोग करते हैं
फ़ंक्शन और व्यवहार का संदर्भ जिसका वर्णन किया जाना है
 अभिनेता कुछ भी हैं जो सिस्टम या उत्पाद के साथ संवाद करते हैं और जो हैं
Sys मंदिर के लिए बाहरी
o चरण दो - उपयोग के मामलों को विकसित करें, जहां प्रत्येक एक का जवाब कतार का एक सेट देता
हैस्टिअन

प्रश्नोत्तरी:
1। एक उपयोग केस आरेख के चार मुख्य घटक क्या हैं?
2। उपयोग के मामले में उपयोग की जाने वाली सूची संबंध।
3। क्या परीक्षण उपयोगी उपयोग के मामलों को खोजने में मदद कर सकते हैं?

सुझाए गए संदर्भ:
1। रोजर एस प्रेसमैन, सॉफ्टवेयर इंजीनियरिंग - एक व्यवसायी का दृष्टिकोण, मैकग्रा -हिल
अंतर्राष्ट्रीय संस्करण
2। बूच, जी। एट अल। एकीकृत मॉडलिंग भाषा उपयोगकर्ता गाइड। अध्याय 15, 18, 27।
Addison -wesley।
3। जैकबसन, आई। एट अल। ऑब्जेक्ट -ऑरिएंटेड सॉफ्टवेयर इंजीनियर आईएनजी: एक उपयोग -
कैससंचालित दृष्टिकोण।
Addison -wesley।
4। फाउलर, एम। यूएमएल डिस्टिल्ड: स्टैंडर्ड ऑब्जेक्ट मॉडलिंग लैंग्वेज के लिए एक संक्षिप्त गाइड।
अध्याय 5। एडिसन वेस्ले।
कोमल
सॉफ्टवेयर इंजीनियरिंग

छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन
जैसा कि पूरा पूछा गया
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण पूरा
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान पूरा
कार्यान्वयनजैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता
और पूर्ण समझ
कार्यान्वयन
जैसा कि पूछा गया
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता
& समझ
का सही उत्तर
सभी प्रश्न

संकाय के हस्ताक्षर:

कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 7
AIM: बेसिक COCOMO मॉडल को लागू करके समस्या को हल करें।
 उद्देश्य:
1। कोकोमो का उपयोग करके परियोजनाओं को वर्गीकृत करें, और प्रयास और विकास का अनुमान लगाएंnt समय की आवश्यकता है
एक परियोजना के लिए।

 सिद्धांत

एक सॉफ्टवेयर प्रोजेक्ट प्राप्त करने के लिए स्रोत कोड की कुछ सौ लाइनें लिखने के बारे में नहीं है एक विशेष उद्देश्य। एक सॉफ्टवेयर परियोजना का दायरा तुलनात्मक रूप से काफी बड़ा है, और इस तरह की परियोजना को पूरा होने में कई साल लग सकते हैं। हालांकि, वाक्यांश "काफी बड़ा" केवल कुछ (संभवतः अस्पष्ट) गुणात्मक जानकारी दे सकता है। किसी भी अन्य विज्ञान के रूप में और इंजीनियरिंग डिस्क ipline, एक परियोजना कितनी जटिल है मापने के लिए दिलचस्पी होगी।परियोजना योजना चरण की प्रमुख गतिविधियों में से एक, इसलिए, विभिन्न का अनुमान लगाना है उचित निर्णय लेने के लिए परियोजना पैरामीटर। कुछ महत्वपूर्ण परियोजना पैरामीटर यह अनुमान लगाया गया है कि इसमें शामिल हैं:

प्रोजेक्ट का आकार: कोड का आकार क्या होगा, लाइनों, फ़ाइलों की संख्या में लिखा गया है, मॉड्यूल?
लागत: सॉफ्टवेयर विकसित करने में कितना खर्च आएगा? एक सॉफ्टवेयर सिर्फ टुकड़े हो सकता है कोड, लेकिन किसी को प्रबंधकों को भुगतान करना पड़ता है, आरएस, और अन्य परियोजना कर्मियों को दे रहा है।
अवधि:क्लाइंट्स को सॉफ्टवेयर देने से पहले यह कब तक होगा?
प्रयास: टीम के सदस्यों से कितना प्रयास करने की आवश्यकता होगी सॉफ़्टवेयर?
इस प्रयोग में हम एस्टेटिंग प्रोजेक्ट मेट्रिक्स के लिए दो तरीकों पर ध्यान केंद्रित करेंगे:
कोकोमो

कोकोमो (रचनात्मक लागत मॉडल) बोहम द्वारा प्रस्तावित किया गया था। उनके अनुसार, वहाँ सॉफ्टवेयर परियोजनाओं की तीन श्रेणियां हो सकती हैं: कार्बनिक, सेमिडेटैच, और एम्बेडेड। सॉफ्टवेयर की विशेषताओं को देखते हुए वर्गीकरण किया जाता है,विकास टीम और पर्यावरण। ये उत्पाद कक्षाएं आमतौर पर आवेदन, उपयोगिता के अनुरूप हैं और सिस्टम कार्यक्रम, क्रमशः। डेटा प्रोसेसिंग प्रोग्राम्स पर विचार किया जा सकता है

आवेदन कार्यक्रम। कंपाइलर, लिंकर्स, यूटिलिटी प्रोग्राम्स की परीक्षा में हैं। ऑपरेटिंग
सिस्टम, रियल -टाइम सिस्टम प्रोग्राम सिस्टम प्रोग्राम के उदाहरण हैं। एक आसानी से हो सकता है
आश्वस्त करें कि यह एक OS को विकसित करने के लिए अधिक समय और प्रयास लेगा
उपस्थिति प्रबंधन प्रणाली।

ओर्गा की अवधारणाएनआईसी, सेमिडेटैच और एम्बेडेड सिस्टम नीचे वर्णित हैं।
कोमल
सॉफ्टवेयर इंजीनियरिंग

कार्बनिक: एक विकास परियोजना को कार्बनिक प्रकार का कहा जाता है, यदि परियोजना से संबंधित है
एक अच्छी तरह से समझे जाने वाले एप्लिकेशन को विकसित करना विकास टीम छोटी है
सदस्यों के पास पूर्व अनुभव है मैं समान प्रकार की परियोजनाओं के साथ काम कर रहा हूं
सेमिडेटैच: एक विकास परियोजना को अर्धवृत्ताकार प्रकार के रूप में वर्गीकृत किया जा सकता है, यदि
टीम में कुछ अनुभवी और अनुभवहीन भी शामिल हैंस्टाफ टीम के सदस्य
विकसित होने के लिए प्रणाली के प्रकार पर कुछ अनुभव हो सकता है
एंबेडेड: एम्बेडेड प्रकार के विकास परियोजना वे हैं, जिसका उद्देश्य एक विकसित करना है
मशीन हार्डवेयर टीम के आकार से संबंधित सॉफ्टवेयर दृढ़ता से आमतौर पर बड़ा होता है
बोहम ने सुझाव दिया कि परियोजना के मापदंडों का अनुमान तीन के माध्यम से किया जाना चाहिए
चरण: बेसिक कोकोमो, इंटरमीडिएट कोकोमो, और पूरा कोकोमो।

मूल कोकोमो मॉडल
बेसिक कोकोमो मॉडल प्रोजेक्ट पैरामेट का एक मोटा अनुमान प्राप्त करने में मदद करता हैरु। यह
निम्नलिखित तरीके से विकास के लिए आवश्यक प्रयास और समय का अनुमान है:
प्रयास = ए * (केडीएसआई) बी पीएम
Tdev = 2.5 * (प्रयास) c महीने

कहाँ

 केडीएसआई किलो डिलीवर किए गए स्रोत में व्यक्त सॉफ्टवेयर का अनुमानित आकार है
निर्देश
 ए, बी, सी सॉफ्टवेयर प्रोजेक्ट की श्रेणी द्वारा निर्धारित स्थिरांक हैं
 प्रयास सॉफ्टवेयर विकास के लिए आवश्यक कुल प्रयास को दर्शाता है, जिसे व्यक्त किया गया है
व्यक्ति के महीने (पीएमएस)
 TDEV सॉफ्टव को विकसित करने के लिए आवश्यक अनुमानित समय को दर्शाता हैमें व्यक्त किया गया है
महीने)
स्थिरांक A, B, C का मान नीचे दिया गया है:

सॉफ्टवेयर प्रोजेक्ट ए बी सी
कार्बनिक 2.4 1.05 0.38
सेमी -डिटैच 3.0 1.12 0.35
एंबेडेड 3.6 1.20 0.32

प्रश्नोत्तरी:
1। मान लें कि एक कार्बनिक प्रकार के सॉफ्टवेयर उत्पाद का आकार 32,000 लाइनें होने का अनुमान लगाया गया है
सोर्स कोड। मान लें कि सॉफ्टवेयर इंजीनियरों का औसत वेतन रु। 15,000/ - प्रति माह। ठानना सॉफ्टवेयर उत्पाद को विकसित करने के लिए आवश्यक प्रयास एND नाममात्र विकास समय।
सुझाए गए संदर्भ:
1) रोजर एस प्रेसमैन, सॉफ्टवेयर इंजीनियरिंग - एक व्यवसायी का दृष्टिकोण, मैकग्रा -हिल अंतर्राष्ट्रीय संस्करण
कोमल
सॉफ्टवेयर इंजीनियरिंग

छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन के रूप में
पूरा पूछा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण पूरा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान पूर्णकार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
पूर्ण समझ
कार्यान्वयन के रूप में

पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
समझ
सभी को सही उत्तर
प्रश्न

संकाय के हस्ताक्षर:

कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 8
उद्देश्य: मॉडलिंग यूएमएल वर्ग आरेख और अनुक्रम आरेख।
उद्देश्य:
1। ग्राफिक रूप से एक वर्ग का प्रतिनिधित्व करते हैं, और संघों के बीचअलग -अलग वर्ग
2। एक प्रणाली में गुजरने वाली गतिविधियों के तार्किक अनुक्रम को पहचानें, और उन्हें चित्रात्मक रूप से प्रतिनिधित्व करें
 सिद्धांत:
वर्ग आरेख
यह इसके स्थैतिक निर्माण [1] के संदर्भ में एक प्रणाली का वर्णन करने के लिए एक चित्रमय प्रतिनिधित्व है।

वर्ग आरेख में तत्व
क्लास आरेख में अपने डेटा सदस्य एस, संचालन और आर elationships के साथ सिस्टम कक्षाएं शामिल हैं कक्षाओं के बीच।
कक्षा
समान डेटा सदस्यों और सदस्य कार्यों वाली वस्तुओं का एक सेट एक सीएलएएस द्वारा वर्णित हैएस। में UML सिंटैक्स, क्लास को तीन डिब्बों के साथ ठोस रूपरेखा आयत द्वारा पहचाना जाता है

वर्ग नाम
एक वर्ग को विशिष्ट रूप से उसके नाम से एक प्रणाली में पहचाना जाता है। एक पाठ्य स्ट्रिंग [2] को वर्ग नाम

के रूप में लिया जाता है। यह
वर्ग आयत में पहले डिब्बे में झूठ।

गुण
एक वर्ग के सभी उदाहरणों द्वारा साझा की गई संपत्ति। यह वर्ग आयत में दूसरे डिब्बे में स्थित है।

संचालन
किसी वर्ग की किसी भी वस्तु के लिए एक कार्रवाई का निष्पादन किया जा सकता है। यह अंतिम सह में
हैमातम
वर्ग आयत में।

उदाहरण

एक शैक्षिक संगठन के लिए एक संरचनात्मक मॉडल बनाने के लिए, „कोर्स„ को एक वर्ग के रूप में माना जा सकता है
संचालन के साथ „कोरसेन नाम„ & „कोर्सिड„ शामिल हैं।
„Removecourse ()„ उस वर्ग के लिए किसी भी वस्तु के लिए प्रदर्शन करने की अनुमति दी।
कोमल
सॉफ्टवेयर इंजीनियरिंग

सामान्यीकरण/विशेषज्ञता
यह बताता है कि एक वर्ग दूसरे वर्ग से कैसे लिया जाता है। व्युत्पन्न वर्ग इसके गुणों को विरासत में मिला है
अभिभावकएस ।

Geometric_shapes वह वर्ग है जो बताता है कि किसी विशेष आकार में कितने पक्ष हैं। त्रिकोण,
चतुर्भुज और पेंटागन वे वर्ग हैं जो ज्यामितीय_शेप वर्ग की संपत्ति को विरासत में रखते हैं।
इसलिए इन वर्गों के बीच संबंध सामान्यीकरण हैं। अब lecilateral_triangle,
Isosceles_triangle और scalene_triangle, ये सभी तीन वर्ग त्रिभुज के गुणों को विरासत में रखते हैं
उनमें से प्रत्येक के रूप में कक्षा में तीन पक्ष हैं। तो, ये त्रिभुज वर्ग के विशेषज्ञता हैं।

संबंध
ईटीएक प्रणाली में Xisting रिश्तों में कक्षाओं के बीच वैध संबंध का वर्णन किया गया है
प्रणाली।

संगठन
यह एक उदाहरण स्तर संबंध है [i] जो दोनों की वस्तुओं के बीच संदेशों का आदान -प्रदान करने की अनुमति देता है

एसोसिएशन के छोर। दो वर्ग के बक्से को जोड़ने वाली एक साधारण सीधी रेखा एक एसोसिएशन का प्रतिनिधित्व करती है।

कोमल
सॉफ्टवेयर इंजीनियरिंग

हम एसोसिएशन को एक नाम दे सकते हैं और दोनों छोर पर भी हम भूमिका नामों को इंगित कर सकते हैं और
आसन्न वर्गों की बहुलता। एसोसिएशनtion uni -directional हो सकता है।

उदाहरण

एक संगठन की एक प्रणाली के लिए संरचना मॉडल में एक कर्मचारी („कर्मचारी) वर्ग का उदाहरण) है हमेशा एक विशेष प्रस्थानकर्ता टी („विभाग) वर्ग का उदाहरण) और एसोसिएशन को सौंपा गया संबंधित वर्गों को जोड़ने वाली लाइन द्वारा दिखाया जा सकता है।

एकत्रीकरण
यह एसोसिएशन का एक विशेष रूप है, जो एक जोड़ी के बीच एक भाग के संबंध का वर्णन करता है कक्षाएं। इसका मतलब है, एक रिश्ते में, जब एक वर्ग कुछ उदाहरण रखता हैसंबंधित वर्ग के इक्के, फिर वह संबंध को एकत्रीकरण के रूप में डिज़ाइन किया जा सकता है।

उदाहरण

एक शहर में एक सुपरमार्केट के लिए, प्रत्येक शाखा उनके पास मौजूद कुछ विभागों को चलाती है। तो, संबंध
कक्षाओं के बीच "शाखा„ और „विभाग को एक एकत्रीकरण के रूप में डिज़ाइन किया जा सकता है। यूएमएल में, यह कर सकता है
अंजीर में दिखाया गया है। नीचे

रचना [i]
यह एकत्रीकरण से एक मजबूत है जो बताता है कि पूरे पूरी तरह से अपने हिस्से का मालिक है। ज़िंदगी भाग का चक्र पूरी तरह से डीएस।

बग़ल में होनामातम

कोमल
सॉफ्टवेयर इंजीनियरिंग

एक शॉपिंग मॉल में एक शहर में विभिन्न स्थानों में कई शाखाएं हैं। द एक्ज़िज़टेंस
शाखाओं की पूरी तरह से शॉपिंग मॉल पर निर्भर करती है जैसे कि यह मौजूद नहीं है, इसकी कोई शाखा नहीं होगी
शहर में लंबे समय तक मौजूद है। इस संबंध को रचना के रूप में वर्णित किया जा सकता है और इसे दिखाया जा सकता है
नीचे

 बहुलता
यह बताता है कि एक वर्ग के उदाहरणों की संख्या कितनी संख्या में है
में एक और वर्गएक एसोसिएशन।
विभिन्न प्रकार की बहुलता के लिए संकेतन:

o अनुक्रम आरेख:
 अनुक्रम आरेख सिस्टम में संदेशों के प्रवाह का प्रतिनिधित्व करता है और भी है
एक घटना आरेख के रूप में कहा जाता है। यह कई गतिशील परिदृश्यों की कल्पना करने में मदद करता है। यह
टिम ई-ऑर्डर किए गए अनुक्रम के रूप में किसी भी दो जीवन रेखा के बीच संचार को चित्रित करता है
घटनाओं में से, जैसे कि इन जीवन रेखाओं ने रन के समय में भाग लिया। UML में, जीवन रेखा है
एक ऊर्ध्वाधर बार द्वारा दर्शाया गया है, जबकि संदेश प्रवाह द्वारा प्रतिनिधित्व किया जाता हैए
ऊर्ध्वाधर बिंदीदार रेखा जो पृष्ठ के निचले भाग में फैली हुई है। यह छिद्रों को छिद्रित करता है
पुनरावृत्तियों के साथ -साथ शाखा भी।
o एक अनुक्रम आरेख का उद्देश्य
1। एक प्रणाली के भीतर सक्रिय वस्तुओं के बीच उच्च -स्तरीय बातचीत को मॉडल करने के लिए।
2। एक उपयोग के मामले को साकार करने वाले सहयोग के अंदर वस्तुओं के बीच बातचीत को मॉडल करने के लिए।
3। यह या तो जेनेरिक इंटरए ctions या बातचीत के कुछ निश्चित उदाहरणों को मॉडल करता है।

कोमल
सॉफ्टवेयर इंजीनियरिंग

o एक अनुक्रम आरेख के नोटेशन
लाइफलाइन
एकअनुक्रम आरेख में व्यक्तिगत प्रतिभागी को एक जीवन रेखा द्वारा दर्शाया गया है। यह है
आरेख के शीर्ष पर स्थित है।

अभिनेता
एक इकाई द्वारा निभाई गई भूमिका जो विषय के साथ बातचीत करती है, उसे एक अभिनेता कहा जाता है।

यह बाहर है
सिस्टम का दायरा। यह भूमिका का प्रतिनिधित्व करता है, जिसमें मानव उपयोगकर्ता और बाहरी शामिल हैं
हार्डवेयर या विषय। एक अभिनेता एक भौतिक इकाई का प्रतिनिधित्व कर सकता है या नहीं कर सकता है,
लेकिन यह विशुद्ध रूप से
एक इकाई के rol e को दर्शाता है। एक अभिनेता या वाइस द्वारा कई अलग -अलग भूमिकाएं निभाई जा
सकती हैंवर्सा।

सक्रियण
यह जीवन रेखा पर एक पतली आयत द्वारा दर्शाया गया है। यह उस समय अवधि का वर्णन करता है जिसमें
एक ऑपरेशन एक तत्व द्वारा किया जाता है, जैसे कि आयत के ऊपर और नीचे
दीक्षा और पूरा होने के समय के साथ जुड़ा हुआ है, प्रत्येक क्रमशः।

कोमल
सॉफ्टवेयर इंजीनियरिंग

संदेशों
संदेश वस्तुओं के बीच बातचीत को दर्शाते हैं और तीर द्वारा दर्शाया जाता है।
वे लाइफलाइन पर अनुक्रमिक क्रम में हैं। का सह पुन:अनुक्रम आरेख है
संदेशों और जीवन रेखाओं द्वारा गठित।
निम्नलिखित संदेशों के प्रकार नीचे दिए गए हैं:
 कॉल संदेश: यह एक की जीवन रेखा के बीच एक विशेष संचार को परिभाषित करता है
इंटरैक्शन, जो यह दर्शाता है कि लक्ष्य लाइफलाइन ने एक ऑपरेशन किया है।

संदेश लौटाता है: यह जीवन रेखा के बीच एक विशेष संचार को परिभाषित करता है
बातचीत जो संबंधित के रिसीवर से सूचना के प्रवाह का प्रतिनिधित्व करती है
कॉलर संदेश।

स्व संदेश: यह वर्णन करता हैएक संचार, विशेष रूप से एक की जीवन रेखा के बीच
बातचीत जो एक ही जीवन रेखा के संदेश का प्रतिनिधित्व करती है, को आमंत्रित किया गया है।

कोमल
सॉफ्टवेयर इंजीनियरिंग

पुनरावर्ती संदेश: पुनरावर्ती उद्देश्य के लिए भेजे गए एक स्व संदेश को पुनरावर्ती कहा जाता है
संदेश। अन्य खराब डीएस में, यह कहा जा सकता है कि पुनरावर्ती संदेश एक विशेष मामला है

स्व संदेश के रूप में यह पुनरावर्ती कॉल का प्रतिनिधित्व करता है।

संदेश बनाएँ: यह एक संचार का वर्णन करता है, विशेष रूप से लाइफेलिन के बीचएक के es
इंटरैक्शन का वर्णन है कि लक्ष्य (लाइफेल INE) को त्वरित किया गया है।

संदेश को नष्ट करें: यह एक संचार का वर्णन करता है, विशेष रूप से एक की जीवन रेखा के बीच
बातचीत जो लक्ष्य के जीवनचक्र को नष्ट करने के अनुरोध को दर्शाती है।

कोमल
सॉफ्टवेयर इंजीनियरिंग

अवधि संदेश: यह विशेष रूप से एक संचार का वर्णन करता है
इंटरैक्शन, जो किसी सिस्टम को मॉडलिंग करते समय संदेश के समय पारित होने का चित्रण करता है।
प्रश्नोत्तरी:
1) एक सेक्क मेंEnce आरेख, एक बॉक्स क्या दर्शाता है? एक धराशायी रेखा क्या दर्शाती है? क्या करता है
बक्से के बीच तीर चित्रित?
2) एक लाइफलाइन पर एक एक्स क्या इंगित करता है?

सुझाए गए संदर्भ:
1) राजिब मॉल, सॉफ्टवेयर इंजीनियरिंग के बुनियादी बातें, भारत का प्रेंटिस हॉल।

2) पंकज जलोट, सॉफ्टवेयर इंजीनियरिंग - एक सटीक दृष्टिकोण विली

छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन के रूप में
पूरा पूछा
कार्यान्वयन के रूप मेंपूछा
समस्या विश्लेषण पूरा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान पूर्ण

कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
पूर्ण समझ
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
समझ
सभी को सही उत्तर
प्रश्न

संकाय के हस्ताक्षर:
कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 9
AIM: VARI डिजाइन करेंसिस्टम के परीक्षण को करने के लिए ous परीक्षण मामलों में और विभिन्न प्रदर्शन करने के लिए
परीक्षण का प्रकार
उद्देश्य: विभिन्न परीक्षण तकनीकों के बारे में पता लगाने और सीखने के लिए और उनका उपयोग करना।

 सिद्धांत:
o सॉफ्टवेयर परीक्षण उपयोगकर्ताओं से एकत्रित आवश्यकताओं के खिलाफ सॉफ्टवेयर का मूल्यांकन है और सिस्टम विनिर्देश। परीक्षण सॉफ्टवेयर में चरण स्तर पर आयोजित किया जाता है
विकास जीवन चक्र या कार्यक्रम कोड में मॉड्यूल स्तर पर। सॉफ्टवेयर परीक्षण में शामिल हैं
सत्यापन और सत्यापन।
ओ नरमबर्तन सत्यापन
 सत्यापन यह जांचने की प्रक्रिया है कि सॉफ्टवेयर उपयोगकर्ता को संतुष्ट करता है या नहीं आवश्यकताएं। यह एसडीएलसी के अंत में किया जाता है। अगर सॉफ्टवेयर मेल खाता है
आवश्यकताएं जिसके लिए इसे बनाया गया था, यह मान्य है।
o सत्यापन सुनिश्चित करता है कि विकास के तहत उत्पाद उपयोगकर्ता के अनुसार है

आवश्यकताएं।
o सत्यापन प्रश्न का उत्तर देता है - "क्या हम उत्पाद विकसित कर रहे हैं जो
इस सॉफ़्टवेयर से उपयोगकर्ता की जरूरत के सभी प्रयासों का प्रयास करता है? ”।
o सत्यापन उपयोगकर्ता आवश्यकताओं पर जोर देता है।o सॉफ्टवेयर सत्यापन
 सत्यापन पुष्टि जी की प्रक्रिया है यदि सॉफ्टवेयर व्यवसाय को पूरा कर रहा है
आवश्यकताओं, और उचित विनिर्देशों का पालन करते हुए विकसित किया जाता है और
कार्यप्रणाली।
 सत्यापन सुनिश्चित करता है कि उत्पाद विकसित किया जा रहा है डिजाइन के अनुसार है
विशेष विवरण।
 सत्यापन प्रश्न का उत्तर देता है - "क्या हम इस उत्पाद को मजबूती से विकसित कर रहे हैं
सभी डिजाइन विनिर्देशों का पालन करें? "
 सत्यापन डिजाइन और सिस्टम विनिर्देशों पर ध्यान केंद्रित करता है।
o लक्ष्यपरीक्षण के हैं -
 त्रुटियां - ये डेवलपर्स द्वारा की गई वास्तविक कोडिंग गलतियाँ हैं। इसके अलावा, एक है
सॉफ्टवेयर और वांछित आउटपुट के आउटपुट में अंतर, एक त्रुटि के रूप में माना जाता है।
कोमल
सॉफ्टवेयर इंजीनियरिंग

 गलती - जब त्रुटि मौजूद होती है तो गलती होती है। एक गलती, जिसे बग के रूप में भी जाना जाता है, एक का परिणाम है
त्रुटि जो प्रणाली विफल हो सकती है।
 विफलता - विफलता को वांछित कार्य करने में असमर्थता ओ एफ कहा जाता है।
विफलता तब होती है जब गलती टी में मौजूद होती हैवह प्रणाली।
o परीक्षण स्तर
 परीक्षण स्वयं एसडीएलसी के विभिन्न स्तरों पर परिभाषित किया जा सकता है। परीक्षण प्रक्रिया
सॉफ्टवेयर विकास के समानांतर चलता है। अगले चरण में कूदने से पहले, ए
चरण का परीक्षण, मान्य और सत्यापित किया जाता है।
 अलग से परीक्षण केवल यह सुनिश्चित करने के लिए किया जाता है कि कोई छिपे हुए कीड़े या नहीं हैं
सॉफ्टवेयर में छोड़े गए मुद्दे। सॉफ्टवेयर विभिन्न स्तरों पर परीक्षण किया जाता है -
o इकाई परीक्षण
कोडिंग करते समय, प्रोग्रामर कार्यक्रम के tha t इकाई पर कुछ परीक्षण करता है
पता है कि क्या यह त्रुटि मुक्त हैई। परीक्षण व्हाइट -बॉक्स परीक्षण दृष्टिकोण के तहत किया जाता है। इकाई परीक्षण में डेवलपर्स को यह तय करने में मदद मिलती है कि कार्यक्रम की व्यक्तिगत इकाइयाँ काम कर रही हैं
प्रति आवश्यकता और त्रुटि मुक्त हैं।
o एकीकरण परीक्षण
यहां तक कि अगर वह सॉफ्टवेयर की इकाइयाँ व्यक्तिगत रूप से ठीक काम कर रही हैं, तो खोजने की

आवश्यकता है
यदि इकाइयाँ यदि एकीकृत एक साथ एक साथ काम करती हैं तो भी त्रुटियों के बिना काम करेंगी। उदाहरण के लिए,
तर्क पासिंग और डेटा अपडेशन आदि।
o सिस्टम परीक्षण
सॉफ्टवेयर को produ के रूप में संकलित किया गया हैसीटी और फिर इसे एक पूरे के रूप में परीक्षण किया जाता है।
 पृष्ठभूमि / तैयारी:
o परीक्षण प्रबंधन उपकरण
o परीक्षण प्रबंधन उपकरण का उपयोग सभी परीक्षण गतिविधि, फास्ट डेटा का ट्रैक रखने के लिए किया जाता है
विश्लेषण, मैनुअल और स्वचालन परीक्षण मामलों, विभिन्न वातावरणों का प्रबंधन, और
योजना बनाएं और मैनुअल परीक्षण भी बनाए रखें।
o परीक्षण प्रबंधन उपकरणों का उपयोग सभी टी परीक्षण गतिविधि, तेजी से डेटा विश्लेषण पर नज़र रखने के लिए किया जाता है,
मैनुअल और ऑटोमेशन टेस्ट के मामलों, विभिन्न वातावरणों और योजना और रखरखाव का प्रबंधन करें
मैनुअल टेस्टीएनजी के रूप में अच्छी तरह से।
o परीक्षण प्रबंधन उपकरण स्वचालन सॉफ्टवेयर के साथ जुड़ा हुआ है। इस प्रकार के
टूल्स में परीक्षण और कई सेट सुविधाओं के लिए विभिन्न स्ट्रैट एगिस थे। परीक्षण के कुछ
प्रबंधन उपकरणों में आवश्यकताओं की मदद से परीक्षण मामले को डिजाइन करने की क्षमता थी।
कोमल
सॉफ्टवेयर इंजीनियरिंग

o यह परीक्षण प्रबंध, शेड्यूलिंग, दोष लॉगिंग, ट्रैकिंग और विश्लेषण के लिए सबसे अच्छा है।
o सबसे अधिक सह -उपयोग किए जाने वाले परीक्षण प्रबंधन उपकरणों में से कुछ इस प्रकार हैं:
o गुणवत्ता प्रतिशतएर
o rth
ओ टेस्टपैड
o टेस्ट मॉनिटर
o प्रैक्टिटेस्ट
 उपकरण / सामग्री की आवश्यकता:
ओ हार्डवेयर:
ओ सॉफ्टवेयर:

o परीक्षण के मामले:
 परीक्षण मामले को शर्तों के एक समूह के रूप में परिभाषित किया गया है जिसके तहत एक परीक्षक निर्धारित करता है
चाहे सॉफ्टवेयर एप्लिकेशन ग्राहक की आवश्यकताओं के अनुसार काम कर रहा हो या

नहीं। टेस्ट केस डिजाइनिंग में पूर्व शर्त, केस नाम, इनपुट शर्तें और शामिल हैं
अपेक्षित परिणाम। एक परीक्षण मामला एक प्रथम स्तर की कार्रवाई है और परीक्षण परिदृश्यों से प्राप्त होता है।
o टेस्ट केस टेम्पलेट
ओ पीआरएक परीक्षण मामला लिखने का उद्देश्य की दक्षता प्राप्त करना है
आवेदन पत्र।
परीक्षा
मामला
आईडी परीक्षण
परिदृश्य परीक्षण चरण अपेक्षित
वास्तविक परिणाम
परिणाम पास/विफल
1
2

उदाहरण
परीक्षा
मामला
आईडी परीक्षण
परिदृश्य परीक्षण चरण अपेक्षित
वास्तविक परिणाम
परिणाम पास/विफल
TU01 चेक
ग्राहक
1 के साथ लॉगिन करें। पर जाएं
साइट http: //demo.guru99.c
ओम उपयोगकर्ता
चाहिए
लॉगिन में लॉगिन करना
के रूप में आवेदन करना
अपेक्षित पास
कोमल
सॉफ्टवेयर इंजीनियरिंग

मान्य डेटा 2। उपयोगकर्ता दर्ज करेंपहचान
3। पासवर्ड दर्ज करें
4। सबमिट पर क्लिक करें
TU02 चेक

ग्राहक
साथ प्रवेश करना
अमान्य डेटा 1। पर जाएं
साइट http: //demo.guru99.c
ओम
2। UserId दर्ज करें
3। पासवर्ड दर्ज करें
4। उपयोगकर्ता सबमिट करें पर क्लिक करें
नहीं करना चाहिए
लॉगिन में लॉगिन करना
के रूप में आवेदन करना
अपेक्षित पास

प्रश्नोत्तरी:
1 WebApp के कौन से तत्व "यूनिट परीक्षण" हो सकते हैं? केवल किस प्रकार के परीक्षण किए जाने चाहिए WebApp तत्वों को एकीकृत करने के बाद?
2 व्हाइट बॉक्स परीक्षण क्या है? विभिन्न कवरेज आधारित परीक्षण रणनीतियों क्या हैं।
3 क्या हैब्लैक बॉक्स परीक्षण?

सुझाए गए संदर्भ:
1 सॉफ्टवेयर परीक्षण: एक शिल्पकार का दृष्टिकोण, पॉल सी। जोर्गेनसेन, तीसरे संस्करण द्वारा
राजिब मॉल, PHI 2014 द्वारा 2 सॉफ्टवेयर इंजीनियरिंग

छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:

कोमल
सॉफ्टवेयर इंजीनियरिंग

रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन के रूप में
पूरा पूछा
कार्यान्वयन के रूप में

पूछा
समस्या विश्लेषण पूरा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकाससमाधान पूर्ण
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
पूर्ण समझ
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
समझ
का सही उत्तर
सभी प्रश्न

संकाय के हस्ताक्षर:

कोमल
सॉफ्टवेयर इंजीनियरिंग

व्यावहारिक - 10
उद्देश्य: इन्फ्रास्ट्रक्चर ऑटोमेशन, कॉन्फ़िगरेशन के लिए DevOps में ओपन -स्रोत उपकरणों का अध्ययन प्रबंध, डेटैनाती स्वचालन, प्रदर्शन प्रबंधन, लॉग प्रबंधन। निगरानी

 उद्देश्य: यह जानने के लिए कि देवो पीएस उपकरण कैसे काम करते हैं।
 सिद्धांत:
DevOps सांस्कृतिक दर्शन, प्रथाओं और उपकरणों का संयोजन है जो एक संगठन को बढ़ाता है
उच्च वेग पर अनुप्रयोगों और सेवाओं को वितरित करने की क्षमता: एक तेजी से उत्पादों को विकसित करना और सुधार करना
पारंपरिक सॉफ्टवेयर डे velopment और इन्फ्रास्ट्रक्चर मैनेजमेंट का उपयोग करने वाले संगठनों की तुलना में पेस
प्रक्रियाएं। यह गति संगठनों को बेट करने में सक्षम बनाती हैएर अपने ग्राहकों की सेवा करता है और अधिक प्रभावी ढंग से प्रतिस्पर्धा करता है
बाजार में।

कैसे काम करता है
एक DevOps मॉडल के तहत, विकास और संचालन टीमों को अब "चुप" नहीं किया जाता है। कभी -कभी, ये दो टीमों को एक एकल टीम में विलय कर दिया जाता है जहां इंजीनियर पूरे आवेदन में काम करते हैं
जीवनचक्र, विकास और परीक्षण से लेकर परिनियोजन तक संचालन तक, और कौशल की एक श्रृंखला विकसित नहीं करें
एक ही फ़ंक्शन तक सीमित।
कुछ देवो पीएस मॉडल में, गुणवत्ता आश्वासन और सुरक्षा टीमें भी मोर बन सकती हैंई कसकर एकीकृत विकास और संचालन के साथ और पूरे अनुप्रयोग जीवनचक्र में। जब सुरक्षा का ध्यान केंद्रित होता है
एक DevOps टीम पर हर कोई, इसे कभी -कभी देवता ECOPS के रूप में संदर्भित किया जाता है।
ये टीमें उन प्रक्रियाओं को स्वचालित करने के लिए प्रथाओं का उपयोग करती हैं जो ऐतिहासिक रूप से मैनुअल और धीमी रही हैं। वे उपयोग करते हैं
एक प्रौद्योगिकी ढेर और टूलींग जो उन्हें जल्दी और मज़बूती से अनुप्रयोगों को संचालित करने और विकसित करने में मदद करता है।
ये उपकरण इंजीनियरों को स्वतंत्र रूप से ish कार्यों को पूरा करने में भी मदद करते हैं (उदाहरण के लिए, कोड ओ को तैनात करनाआर

कोमल
सॉफ्टवेयर इंजीनियरिंग

इंफ्रास्ट्रक्चर का प्रावधान) जो आम तौर पर अन्य टीमों से मदद की आवश्यकता होती है, और यह आगे एक टीम के वेग को बढ़ाता है।
DevOps मायने क्यों रखता है

सॉफ्टवेयर और इंटरनेट ने खरीदारी से लेकर मनोरंजन तक दुनिया और उसके i ndustries को बदल दिया है
बैंकिंग के लिए। सॉफ्टवेयर अब केवल एक व्यवसाय का समर्थन नहीं करता है; बल्कि यह एक अभिन्न अंग

बन जाता है
एक व्यवसाय का हर हिस्सा। कंपनियां सॉफ्टवेयर डे के माध्यम से अपने ग्राहकों के साथ बातचीत करती हैंऑनलाइन के रूप में लिवर
सेवाओं या अनुप्रयोगों और सभी प्रकार के उपकरणों पर। वे ऑपरेशनल बढ़ाने के लिए सॉफ्टवेयर का भी उपयोग करते हैं
मूल्य श्रृंखला के प्रत्येक भाग को बदलकर, जैसे कि रसद, संचार, और
संचालन। इसी तरह से कि भौतिक वस्तुओं की कंपनियों ने ट्रांसफ़र किया कि वे कैसे डिजाइन, निर्माण करते हैं, और
20 वीं शताब्दी के दौरान औद्योगिक स्वचालन का उपयोग करके उत्पाद वितरित करें, आज की दुनिया में कंपनियां
यह बदलना चाहिए कि वे सॉफ्टवेयर का निर्माण और वितरण कैसे करते हैं।
देवप्स प्रैक्टिस

सीontinuous एकीकरण
निरंतर एकीकरण एक सॉफ्टवेयर विकास अभ्यास है जहां डेवलपर्स नियमित रूप से अपने कोड को मर्ज करते हैं
एक केंद्रीय भंडार में परिवर्तन, जिसके बाद स्वचालित बिल्ड और परीक्षण चलाए जाते हैं। के प्रमुख लक्ष्य निरंतर एकीकरण बग्स को जल्दी खोजने और संबोधित करने, सॉफ्टवेयर योग्यता में सुधार करने और कम करने के लिए हैं
नए सॉफ़्टवेयर अपडेट को मान्य और जारी करने में समय लगता है।

निरंतर वितरण
निरंतर वितरण एक सॉफ्टवेयर विकास अभ्यास है जहां कोड परिवर्तन ऑटो हैंमैटिक रूप से बनाया गया, परीक्षण किया, और उत्पादन के लिए एक रिलीज के लिए तैयार किया। यह सभी को तैनात करके निरंतर एकीकरण पर EXP &
बिल्ड स्टेज के बाद कोड एक परीक्षण वातावरण और/या उत्पादन वातावरण में बदलता है। कब
निरंतर डिलीवरी ठीक से लागू की जाती है, डेवलपर्स के पास हमेशा एक परिनियोजन होगा -डी बिल्ड होगा आर्टिफ़ैक्ट टी हैट एक मानकीकृत परीक्षण प्रक्रिया से गुजरा है।

माइक्रोसर्विसिस
MicroServices आर्किटेक्चर एक सेट O के रूप में एकल एप्लिकेशन बनाने के लिए एक डिज़ाइन दृष्टिकोण हैएफ छोटा
सेवाएं। प्रत्येक सेवा अपनी प्रक्रिया में चलती है और एक कुएं के माध्यम से अन्य सेवाओं के साथ संचार करती है -
एक हल्के तंत्र का उपयोग करके परिभाषित इंटरफ़ेस, आमतौर पर एक HTTP -आधारित एप्लिकेशन प्रोग्रामिंग
इंटरफ़ेस (एपीआई)। MicroServices व्यावसायिक क्षमताओं के आसपास निर्मित होते हैं; प्रत्येक सेवा को एक एकल में स्कोप किया जाता है

उद्देश्य। आप माइक्रोसर्विस और लिखने के लिए अलग -अलग फ्रेमवो आरके या प्रोग्रामिंग भाषाओं का उपयोग कर सकते हैं और
उन्हें स्वतंत्र रूप से, एक ही सेवा के रूप में, या सेवाओं के समूह के रूप में तैनात करें।
कोमल
सॉफ्टवइंजीनियरिंग हैं (3161605)

कोड के रूप में अवसंरचना
कोड के रूप में इन्फ्रास्ट्रक्चर एक अभ्यास है जिसमें बुनियादी ढांचे का प्रावधान किया जाता है और कोड का उपयोग करके प्रबंधित किया जाता है और
सॉफ्टवेयर विकास तकनीक, जैसे कि संस्करण नियंत्रण और निरंतर एकीकरण। बादल की एपीआई -
संचालित मॉडल डेवलपर्स और सिस्टम प्रशासकों को बुनियादी ढांचे के साथ बातचीत करने में सक्षम बनाता है
प्रोग्रामेटिक रूप से, और पैमाने पर, मैन्युअल रूप से एस एट अप और संसाधनों को कॉन्फ़िगर करने की आवश्यकता के बजाय। इस प्रकार,
इंजीनियर बुनियादी ढांचे के साथ इंटरफेस कर सकते हैंURE कोड -आधारित टूल का उपयोग करना और एक तरीके से बुनियादी ढांचे का इलाज करना
वे कैसे एप्लिकेशन कोड का इलाज करते हैं। क्योंकि वे कोड, इन्फ्रास्ट्रक्चर और सर्वर द्वारा परिभाषित हैं
क्या क्किकल y को मानकीकृत पैटर्न का उपयोग करके तैनात किया जा सकता है, नवीनतम पैच और संस्करणों के साथ अपडेट किया गया, या
दोहराए जाने वाले तरीकों से डुप्लिकेट किया गया।
विन्यास प्रबंधन
डेवलपर्स और सिस्टम एडमिनिस्ट्रेटर ऑपरेटिंग सिस्टम और होस्ट कॉन्फ़िगरेशन को स्वचालित करने के लिए कोड का उपयोग करते हैं,
ऑपरेशन अल कार्य, और बहुत कुछ। कोड का उपयोग कॉन्फ़िगरेशन बनाता हैn दोहराव और मानकीकृत परिवर्तन।
यह डेवलपर्स और सिस्टम प्रशासकों को मैन्युअल रूप से ऑपरेटिंग सिस्टम, सिस्टम को कॉन्फ़िगर करने से मुक्त करता है
एप्लिकेशन, या सर्वर सॉफ्टवेयर।
कोड के रूप में नीति
Infrastruc ture और इसके कॉन्फ़िगरेशन को क्लाउड के साथ संहिताबद्ध करने के साथ, संगठन मॉनिटर और लागू कर सकते हैं
गतिशील रूप से और पैमाने पर अनुपालन। इस प्रकार कोड द्वारा वर्णित इन्फ्रास्ट्रक्चर को ट्रैक किया जा सकता है,
मान्य, और एक स्वचालित तरीके से पुन: कॉन्फ़िगर किया गया। यह संगठनों को शासन करने के लिए इसे आसान बनाता हैसंसाधनों पर परिवर्तन और यह सुनिश्चित करें कि सुरक्षा उपायों को वितरित तरीके से ठीक से लागू किया गया है
(जैसे सूचना सुरक्षा या PCI -DSS या HIPAA का अनुपालन)। यह टीमों को एक के भीतर अनुमति देता है

गैर -आर -वैलेंट संसाधनों के बाद से उच्च आर वेग में स्थानांतरित करने के लिए संगठन को स्वचालित रूप से चिह्नित किया जा सकता है
आगे की जांच या यहां तक कि स्वचालित रूप से अनुपालन में वापस लाया गया।
निगरानी और लॉगिंग
संगठन मैट्रिक्स की निगरानी करते हैं और यह देखने के लिए कि कैसे अनुप्रयोग और बुनियादी ढांचा पूर्ण ormance यह देखने के लिएप्रभाव डालता है
उनके उत्पाद के अंतिम उपयोगकर्ता का अनुभव। कैप्चर करने, वर्गीकृत करने और फिर डेटा और लॉग का विश्लेषण करके
अनुप्रयोगों और बुनियादी ढांचे से उत्पन्न, संगठन समझते हैं कि कैसे परिवर्तन या अद्यतन प्रभाव प्रभाव डालते हैं
उपयोगकर्ता, समस्याओं या अप्रत्याशित परिवर्तनों के मूल कारणों में अंतर्दृष्टि को बहाते हैं। सक्रिय निगरानी सेवाओं को 24/7 और आवेदन और बुनियादी ढांचे के रूप में उपलब्ध होना चाहिए
अद्यतन आवृत्ति बढ़ जाती है। अलर्ट बनाना या रियल -टाइम विश्लेषण करनाइस दा टा भी मदद करता है संगठन अपनी सेवाओं की अधिक निगरानी करते हैं।
संचार और सहयोग
एक संगठन में संचार और सहयोग में वृद्धि के प्रमुख सांस्कृतिक पहलुओं में से एक है
Devops। सॉफ्टवेयर वितरण प्रक्रिया के DevOps टूलिंग और स्वचालन का उपयोग स्थापित करता है शारीरिक रूप से वर्कफ़्लोज़ और विकास की जिम्मेदारियों को एक साथ लाकर सहयोग और
संचालन। इसके शीर्ष पर निर्माण, इन टीमों ने इन्फोर के आसपास मजबूत सांस्कृतिक मानदंडों को निर्धारित कियामेशन शेयरिंग और
चैट एप्लिकेशन, इश्यू या प्रोजेक्ट ट्रैकिंग सिस्टम के उपयोग के माध्यम से संचार की सुविधा, और
कोमल
सॉफ्टवेयर इंजीनियरिंग

विकिस। यह डेवलपर्स, संचालन और यहां तक कि अन्य टीमों में भी संचार को गति देने में मदद करता है विपणन या बिक्री, संगठन के सभी हिस्सों को लक्ष्यों और परियोजनाओं पर अधिक निकटता से अनुमति देता है।
Devops उपकरण
DevOps मॉडल टीमों को तेजी से और मज़बूती से तैनात करने और नवाचार करने में मदद करने के लिए प्रभावी टूलिंग पर निर्भर करता है
उनके व्यापारr ग्राहक। ये उपकरण मैनुअल कार्यों को स्वचालित करते हैं, टीमों को पैमाने पर जटिल वातावरण का प्रबंधन करने में मदद करते हैं,
और इंजीनियरों को उच्च वेग के नियंत्रण में रखें जो DevOps द्वारा सक्षम है। AWS सेवाएं प्रदान करता है यह DevOps के लिए डिज़ाइन किया गया है और जो पहले AWS क्लाउड के साथ उपयोग के लिए बनाया गया है। ये सेवाएं मदद करती हैं
आप ऊपर वर्णित DevOps प्रथाओं का उपयोग करते हैं।
+प्रश्नोत्तरी:

1 DevOps कार्यान्वयन के साथ क्या चुनौतियां हैं?
2 DevOps क्या है? यह काम किस प्रकार करता है? DevOps सिद्धांत और सर्वोत्तम प्रथाएं क्या हैं?
3DevOps जीवनचक्र के 7cs की व्याख्या करें।

सुझाए गए संदर्भ:
1 दीपक गायकवाड़, वायरल थाककर, प्रैक्टिशनर के व्यू के टूल्स, विली से डिवोप्स टूल।
2 द देवप्स हैंडबुक - जीन किम एट। अल।
छात्रों द्वारा उपयोग किए जाने वाले संदर्भ:
रूब्रिक बुद्धिमान निशान प्राप्त:
रूब्रिक्स 1 2 3 4 5 कुल
निशान पूर्ण
कार्यान्वयन के रूप में
पूरा पूछा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण पूरा
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान पूर्ण
कार्यान्वयन के रूप में
पूछासमस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
पूर्ण समझ
कार्यान्वयन के रूप में
पूछा
समस्या विश्लेषण
का विकास
समाधान
अवधारणा स्पष्टता और
समझ
सभी को सही उत्तर
प्रश्न

संकाय के हस्ताक्षर: