

سوال ۱

چرا دستیاران آموزشی درس مهندسی نرم‌افزار به این نتیجه رسیده‌اند که درس، پروژه یا تمرین عملی نداشته باشد؟

جواب سوال ۱

چند علت ممکن است وجود داشته باشد که این‌جا به شرح هر کدام از این علت‌ها و توضیح آن‌ها می‌پردازیم:

الف) هدف اصلی یادگیری این درس: ممکن است از سمت استاد درس و دستیاران، هدف این درس درک مفاهیم مهندسی نرم‌افزار و افزایش مهارت تحلیل دانشجو باشد. پس شاید پروژه‌ی عملی در این راستا نباشد و وجود تمرین تئوری و مفهومی به این هدف کمک بیشتری کنند. در واقع ممکن است هدف‌گذاری استاد درس و دستیاران به این شکل باشد که در پایان این درس، یادگیری مفاهیم و مهارت تحلیل مهم باشد و نه توانایی پیاده‌سازی، برای همین منطقی‌ست که پروژه یا تمرین عملی در سیاست‌های درس جایی نداشته باشند.

ب) زمان کم: فشرده بودن برنامه درسی دانشجویان و زمان کم هر ترم باعث می‌شود که پروژه‌های عملی با کیفیت خوب و کامل قابل اجرا نباشند و هماهنگی بین افراد گروه به خوبی انجام نگیرد. برای همین شاید منطقی‌ست که پروژه‌های عملی که به‌خصوص در این درس، زمان زیادی نیاز دارند برای این‌که همه‌ی افراد در آن درگیر شوند، حذف شود.

ج) ساختار برنامه‌ی درسی: گاهی اوقات برنامه‌ی درسی به گونه‌ای طراحی شده است که تمرینات عملی در دروس دیگر یا در سطوح بالاتری از تحصیل تعبیه شده‌اند.

د) فرضیه‌های پیش‌زمینه‌ای: ممکن است دستیاران آموزشی فرض کنند که دانشجویان قبلاً مهارت‌های عملی لازم را در دروس دیگر یا از طریق تجربیات شخصی کسب کرده‌اند، برای همین اصلاً یادگیری مبحث جدیدی نمی‌تواند برایشان داشته باشد بخش عملی این درس.

ه) ارزیابی و سنجش: ممکن است سیستم ارزیابی دانشگاه تمرکز بیشتری بر روی امتحانات کتبی داشته باشد و بر این باور باشد که تمرینات عملی به سادگی قابل ارزیابی و نمره‌دهی نیستند.

در هر صورت، اینکه آیا تصمیم برای عدم شامل کردن پروژه‌ها و تمرینات عملی در این درس، تصمیم درستی است یا خیر، به شرایط خاص کلاس و اهداف آموزشی تیم دستیاران و استاد درس بستگی دارد. با این‌که طبق بیان کتاب پرسمن، بدون تجربه‌ی عملی، دانشجویان نمی‌توانند دانش نظری را به طور کامل درک کنند و آن را در محیط‌های واقعی به کار گیرند، منطقی‌ست که پس از آگاهی از هدف‌گذاری تیم تدریس، به بررسی این موارد بپردازیم و تحلیل کنیم.

سوال ۲

تفاوت بین «مهندس نرم‌افزار» و «توسعه‌دهنده نرم‌افزار» چیست؟

جواب سوال ۲

فعالیت‌ها

توسعه‌دهنده نرم‌افزار بخش مشخصی از یک نرم‌افزار یا سیستم کامپیوتری را توسعه می‌دهد، ولی مهندس نرم‌افزار، برنامه‌ریزی، طراحی، توسعه و ساخت کامل یک نرم‌افزار یا سیستم کامپیوتری را بر عهده دارد و همچنین بر اساس اصول مهندسی و معیارهای کیفی، به مدیریت پروژه، تضمین کیفیت، تخمین هزینه و جدول‌بندی زمانی می‌پردازد.

مهارت‌ها

مهارت‌های توسعه‌دهنده نرم‌افزار

دانش زبان‌های برنامه‌نویسی فرانت‌اند و بک‌اند، تست کد، استفاده از ابزارهای نرم‌افزاری جهت توسعه برنامه‌های کاربردی، و درک خوبی از فرآیندهای توسعه چابک و یکپارچه‌سازی مداوم.

مهارت‌های مهندس نرم‌افزار

دانش گسترده از زبان‌های برنامه‌نویسی فرانت‌اند و بک‌اند، دیباگ نرم‌افزار، ایجاد ابزارهای نرم‌افزاری، دانش معماری نرم‌افزار، مهارت همکاری، توانایی درک و تحلیل نیازمندی‌های کسب‌وکار و تبدیل آن‌ها به مشخصات فنی، و همچنین مهارت‌های مربوط به مدیریت پروژه و ارتباط با مشتری.

همکاری با دیگران

توسعه‌دهنده معمولاً به طور مستقل کار می‌کند و بعضاً با توسعه‌دهنده‌های دیگر همکاری می‌کند. مهندس نرم‌افزار در یک محیط کاملاً مشارکتی کار می‌کند و با سایر مهندسان، علاوه بر آن با تیم‌های دیگر مثل محصول، طراحی، تضمین کیفیت، و حتی مشتریان برای تضمین اینکه محصول نهایی نیازهای آن‌ها را برآورده می‌کند، مشارکت دارد.

نتیجه‌گیری

به طور کلی توسعه‌دهنده روی یک بخش از سیستم یا برنامه تمرکز می‌کند در حالی که مهندس نرم‌افزار بر کل پروژه مسئولیت دارد. (می‌توانیم بگوییم انگار توسعه‌دهنده روی هر کدام از طبقات تمرکز دارد، ولی مهندس نرم‌افزار به کل ساختمان و ارتباط و تاثیر تمام طبقات به یکدیگر و در نهایت، کیفیت کل ساختمان توجه دارد، و به نوعی نقش معمار را در پروسه ساخت نرم‌افزار ایفا می‌کند).

سوال ۳

ابتدای فصل اول کتاب پرسمن، داستانی از حضور آقای پرسمن در یک شرکت بازسازی آورده شده است. این داستان را بخوانید. به نظرتان چرا آقای پرسمن این داستان را در ابتدای کتاب خود آورده است؟

جواب سوال ۳

در یک شرکت بازی سازی است. این داستان به خواننده نشان می دهد که آقای پرسمن در عمل به مشکلات و چالش های مربوط به توسعه نرم افزار برخورد کرده است و تجربه های خود را با خواننده به اشتراک می گذارد.

آقای پرسمن این داستان را در ابتدای کتاب آورده است تا خواننده را با مفاهیم و مسائل مربوط به مهندسی نرم افزار آشنا کند. این داستان می تواند به خواننده نشان دهد که توسعه نرم افزار چقدر پیچیده و چالش برانگیز است و نیاز به روش ها و رویکردهای مناسب دارد. همچنین، این داستان می تواند از خواننده خواسته هایی مانند تفکر سیستمی، تحلیل و طراحی نرم افزار، مدیریت پروژه و کیفیت نرم افزار را درک کند.

به طور کلی، این داستان می تواند به خواننده کمک کند تا با مفاهیم اساسی مهندسی نرم افزار آشنا شود و درک بهتری از محتوای کتاب پیدا کند.

برای بررسی جزئی تر دلیل استفاده از این داستان در ابتدای کتاب به تحلیل جزئیات آن می پردازیم:

اهمیت مهندسی نرم افزار: این داستان نشان می دهد که حتی در صنایعی که شاید به ظاهر نیازی به رویکردهای سختگیرانه ی مهندسی نرم افزار ندارند، مانند صنعت بازی های ویدئویی، استفاده از تکنیک ها و اصول مهندسی نرم افزار ضروری است.

تطبیق پذیری مهندسی نرم افزار: داستان بر لزوم انطباق و تطبیق اصول مهندسی نرم افزار با نیازهای خاص یک شرکت یا یک پروژه تأکید دارد، نشان داده می شود که چگونه می توان اصول کلی را برای رسیدن به اهداف خاص، شخصی سازی کرد.

پیچیدگی های فزاینده در توسعه نرم افزار: با افزایش پیچیدگی های بازی ها و کوتاه شدن چرخه های توسعه، نیاز به رویکردهای منضبط تر در توسعه نرم افزار بیشتر شده است. این داستان بر اهمیت استفاده از رویکردهای مهندسی نرم افزار برای موفقیت در چنین محیط هایی تأکید می کند.

تأکید بر نقش «کریتیوها»: داستان به اهمیت تعامل بین تیم های خلاق و توسعه دهندگان فنی اشاره می کند و اینکه چگونه نیازهای خلاقانه باید به زبان فنی ترجمه شوند تا امکان پذیر شدن توسعه واقعی فراهم آید.

در نهایت، با قرار دادن این داستان در ابتدای کتاب، آقای پرسمن سعی دارد ارتباط عمیقی بین نظریه مهندسی نرم افزار و کاربرد عملی آن در صنعت ایجاد کند، و ما را تشویق کند که به این ارتباط اهمیت دهیم و از مفاهیم کتاب برای حل مسائل واقعی استفاده کنیم.

سوال ۴

چرا نیازمندی های پروژه این قدر تغییر می کنند؟ آیا مشتری نمی داند چه می خواهد؟!

جواب سوال ۴

تغییرات در نیازمندی های پروژه معمولاً به دلیل چندین عامل مختلف رخ می دهند.

یکی از این عوامل می تواند عدم دقت در تعریف نیازمندی ها باشد. در ابتدای یک پروژه، مشتری ممکن است نتواند نیازمندی های خود را به طور کامل و دقیق تعریف کند. این ممکن است به دلیل عدم آگاهی کامل از قابلیت ها و محدودیت های سیستم باشد یا به دلیل عدم تجربه در حوزه فناوری اطلاعات. بنابراین، در طول زمان و با پیشرفت پروژه، مشتری ممکن است به نحوه بهتری بتواند نیازمندی های خود را تعریف کند و تغییرات لازم را اعمال کند.

عوامل دیگری مانند تغییر در شرایط کسب و کار، رقابت با سایر سازمان‌ها، تغییرات در فناوری و نیازهای جدید مشتریان نیز می‌توانند باعث تغییر در نیازمندی‌های پروژه شوند. در واقع، در دنیای امروز، بازار و تکنولوژی به سرعت تغییر می‌کنند. آنچه امروز نیاز است، ممکن است چند ماه دیگر منسوخ شود. شرکت‌ها برای بقا در بازار رقابتی باید خود را با این تغییرات هماهنگ کنند، که این امر می‌تواند به تغییر نیازمندی‌ها منجر شود.

همچنین، در برخی موارد، مشتری ممکن است در ابتدا نتواند تمامی جزئیات پروژه را پیش‌بینی کند و تغییرات لازم را در طول زمان اعمال کند. در واقع، فیدبک کاربران نهایی پس از مشاهده‌ی نسخه‌های اولیه‌ی محصول هم می‌تواند منجر به تغییرات در نیازمندی‌ها شود.

همچنین ممکن است محدودیت‌هایی در زمینه‌ی تکنولوژی یا معماری سیستم وجود داشته باشد که تنها در حین توسعه‌ی محصول مشخص شوند و نیازمند تغییر در نیازمندی‌ها باشند.

بعضاً حتی تغییر در قوانین دولتی یا صنعتی نیز می‌تواند منجر به تغییر نیازمندی‌ها شود.

به طور کلی، تغییرات در نیازمندی‌های پروژه نشان از یک فرآیند تکاملی و تعاملی است که در طول زمان بهبود می‌یابد. این تغییرات معمولاً نشان از این دارند که مشتری بهتر متوجه نیازهای خود می‌شود و سعی می‌کند تا بهترین نتیجه را از پروژه بگیرد.

سوال ۵

آیا «متدولوژی ایجاد نرم‌افزار» همان «فرآیند ایجاد نرم‌افزار» است؟ از پاسخ خود با جزئیات و تفصیل دفاع کنید. در «مدل فرآیند عمومی ایجاد نرم‌افزار» در کتاب پرسمن، پنج «فعالیت چارچوبی» معرفی می‌شود که یکی از آن‌ها مدل‌سازی است.

در این فعالیت به صورت کلی چه اقداماتی انجام می‌شود؟ امروزه در عمل چه زبان مدل‌سازی استاندارد شده است؟ توضیح دهید که خروجی‌های فعالیت مدل‌سازی، چگونه و با چه هدفی در فعالیت بعدی از فرآیند عمومی ایجاد نرم‌افزار مورد استفاده قرار می‌گیرد.

آیا فعالیت مدل‌سازی با اصل ششم از اصول دوازده‌گانه‌ی چابک در تناقض است؟ از پاسخ خود با مثال دفاع کنید.

جواب سوال ۵

متدولوژی ایجاد نرم‌افزار: مجموعه‌ای از رویه‌ها، تکنیک‌ها، ابزارها و رویکردهای استاندارد است که برای تولید نرم‌افزار استفاده می‌شود. این رویکردها می‌توانند از مراحل تحقیق و توسعه تا تست و نگهداری نرم‌افزار کشیده شود.

فرآیند ایجاد نرم‌افزار: مرحله‌ای است که یک نرم‌افزار از ابتدای توسعه تا انتشار و نگهداری طی می‌کند. هر متدولوژی ممکن است یک فرآیند خاص را برای توسعه نرم‌افزار تعریف کند.

پس، فرآیند ایجاد نرم‌افزار یکی از جزءهای متدولوژی است.

فرآیند ایجاد نرم‌افزار و متدولوژی ایجاد نرم‌افزار اگرچه به نظر می‌رسد از نظر مفهومی به یکدیگر نزدیک هستند و گاهی اوقات به جای یکدیگر استفاده می‌شوند، اما تفاوت‌های اساسی با یکدیگر دارند.

تفاوت‌ها

- تمرکز فرآیندی نسبت به چارچوب متدولوژیک:
فرآیند ایجاد نرم‌افزار ممکن است شامل مجموعه‌ای از قدم‌های مشخص برای توسعه نرم‌افزار باشد، مانند نیازسنجی، طراحی، پیاده‌سازی، آزمایش و تحویل.
متدولوژی ایجاد نرم‌افزار، از سوی دیگر، شامل فرآیند مذکور به همراه فلسفه‌ها، ابزارها، روش‌ها و بهترین شیوه‌هایی است که چگونگی اجرای هر کدام از این قدم‌ها را تعیین می‌کند. به عنوان مثال، متدولوژی چابک تاکید بر توسعه تدریجی، همکاری نزدیک با مشتری و تطبیق‌پذیری دارد.
- ابزارها و تکنیک‌ها:
یک فرآیند ممکن است استفاده از تکنیک‌های خاصی مانند UML برای طراحی یا JUnit برای آزمایش را پیشنهاد دهد.
یک متدولوژی ممکن است فراتر از تکنیک‌های مشخص برای طراحی و آزمایش رفته و فرهنگ سازمانی، نگرش‌ها، ارزش‌ها و اصولی را که باید در تمامی جنبه‌های توسعه نرم‌افزار رعایت شوند، معرفی کند.
- مقیاس و دامنه:
فرآیندها معمولاً کوچکتر و بیشتر به جنبه‌های عملیاتی توسعه نرم‌افزار مربوط می‌شوند.
متدولوژی‌ها ممکن است در یک دامنه وسیع‌تر و با دیدگاهی جامع‌تر به مدیریت پروژه، استراتژی‌های ارتباطی، آموزش و توسعه تیم، و ملاحظات استراتژیک کلان نگاه کنند.

مثال عینی: Scrum به عنوان متدولوژی

فرآیند Scrum ممکن است به سری از اسپرینت‌های دو هفته‌ای اشاره کند که در هر یک اهداف کوتاه‌مدت تعیین و دنبال می‌شوند.
متدولوژی Scrum، این فرآیند را در چارچوبی از قواعد، نقش‌ها (مانند Scrum Master و Product Owner)، جلسات (مانند دیلی استنداپ) و ابزارها (مانند کانبان برد) قرار می‌دهد و این‌ها همگی به همراه مجموعه‌ای از ارزش‌های کلیدی مانند اعتماد، شفافیت و تعهد مطرح می‌شوند.

مدل‌سازی

مدل‌سازی در زمینه توسعه نرم‌افزار، فرآیندی است برای ایجاد یک مدل مفهومی از یک سیستم کامپیوتری که شامل نرم‌افزار و گاهی اوقات سخت‌افزار مرتبط با آن می‌باشد. مدل‌ها می‌توانند ساختار، رفتار و نحوه تعامل اجزای سیستم با یکدیگر و با کاربران را نشان دهند. در فرآیند مدل‌سازی، معمولاً اقدامات زیر صورت می‌گیرد:

- تعریف نیازمندی‌ها: درک و تعریف دقیق نیازمندی‌های سیستم که باید توسط مدل پوشش داده شوند.
- انتخاب روش مدل‌سازی: تعیین اینکه از چه نوع مدل‌سازی استفاده شود، مانند مدل‌های انتزاعی، مدل‌های داده، مدل‌های رفتاری، یا مدل‌های تعاملی.
- تعریف مفاهیم: مشخص کردن مفاهیم کلیدی و موجودیت‌های مورد نیاز برای مدل، مانند کلاس‌ها، شیء‌ها، عملیات، فرایندها و تعاملات.
- طراحی مدل: استفاده از ابزارها و نمادهای استاندارد برای ترسیم مدل، مثل نمودارهای UML یا BPMN

- تحقق و اعتبارسنجی: ساختن یک نسخه اولیه از مدل و اعتبارسنجی آن برای اطمینان از دقت و کارایی در نمایش مفاهیم و روابط واقعی.
- تکرار و بهبود: بازبینی و بهبود مدل بر اساس بازخورد و یافته‌های جدید برای اطمینان از دقت و کامل بودن مدل.
- مستندسازی: تهیه مستندات کامل و دقیق از مدل و توضیحاتی در مورد چگونگی تعامل اجزاء مختلف.
- تست و شبیه‌سازی: استفاده از مدل برای تست سناریوهای مختلف و شبیه‌سازی رفتار سیستم قبل از پیاده‌سازی واقعی.
- انتقال مدل به طراحی: تبدیل مدل‌های تایید شده به معماری‌ها و طراحی‌هایی که می‌توان بر اساس آن‌ها نرم‌افزار را پیاده‌سازی کرد.

فرایند مدل‌سازی به توسعه‌دهندگان کمک می‌کند تا یک درک مشترک از سیستم و نیازمندی‌های آن پیدا کنند و همچنین به انتقال دانش در میان تیم و ذینفعان کمک می‌کند. همچنین این فرایند می‌تواند در کاهش خطاها و سوء تفاهمات در مراحل بعدی توسعه نرم‌افزار مفید باشد.

فعالیت مدل‌سازی در فرآیند عمومی ایجاد نرم‌افزار

در فعالیت مدل‌سازی، نیازها و مشخصات پروژه به صورت دقیق و واضح مدل‌سازی می‌شوند. این مدل‌ها می‌توانند شامل نمودارهای کلاس، نمودارهای فرآیند، و نمودارهای توالی باشند. زبان مدل‌سازی استاندارد شده امروزه UML یا Unified Modeling Language است. این زبان، یکی از زبان‌های استاندارد برای مدل‌سازی نرم‌افزار است که برای توصیف و طراحی سیستم‌ها استفاده می‌شود.

خروجی‌های فعالیت مدل‌سازی، به عنوان نقشه راه برای برنامه‌نویسان و توسعه‌دهندگان عمل می‌کنند و به آن‌ها کمک می‌کند تا با دیدی روشن‌تر، به طراحی و پیاده‌سازی سیستم بپردازند.

تناقض مدل‌سازی با اصول چابک

متدولوژی‌های نسل سومی، مانند Rational Unified Process (RUP) بر فرایندهای ساختاریافته و مرحله‌ای تأکید زیادی داشتند و به طور معمول شامل مراحل مشخصی بودند که باید به ترتیب دنبال می‌شدند. در این متدولوژی‌ها، مدل‌سازی نقش محوری داشت و اغلب با استفاده از ابزارهای مهندسی نرم‌افزار و نمودارهایی مانند UML انجام می‌شد. هدف از مدل‌سازی، ایجاد یک نمایش دقیق و کامل از سیستم قبل از آغاز برنامه‌نویسی واقعی بود، به طوری که تمام جنبه‌ها و پیچیدگی‌های سیستم در مدل‌ها در نظر گرفته شده باشد.

Rational Unified Process (RUP)

RUP به عنوان یک متدولوژی تکراری و تدریجی، به توسعه‌دهندگان این امکان را می‌داد که بخش‌های مختلفی از نرم‌افزار را در مراحل مختلف توسعه بسازند و بهبود ببخشند، که هر کدام ممکن بود شامل مدل‌سازی باشد. در RUP، مدل‌سازی به عنوان یک ابزار برای کاهش ابهامات، پیش‌بینی مشکلات احتمالی و تسهیل ارتباط بین اعضای تیم در نظر گرفته می‌شد.

ظهور رویکردهای Agile

با ظهور رویکردهای Agile، تمرکز از مدل سازی و مستندسازی گسترده به سمت توسعه سریع و انعطاف پذیر منتقل شد. اصول Agile به کارآیی و سادگی تأکید دارند، و اصل ششم منشور Agile بیان می کند که روش ارتباطی مؤثرترین و کارآمدترین روش انتقال اطلاعات به تیم توسعه و درون آن است مکالمه رو در رو است. این موضوع به بحث و مناقشه ای در جامعه توسعه نرم افزار منجر شد، که آیا مدل سازی به طور کامل از بین خواهد رفت یا خیر.

تعادل بین مدل سازی و گفتگوی رو در رو

در واقعیت، Agile نفی کامل مدل سازی را مد نظر ندارد، بلکه به دنبال یافتن تعادل مناسب بین مدل سازی و ارتباطات غنی است. در این رویکرد، مدل سازی هنوز هم می تواند به عنوان ابزاری برای فکر کردن از طریق مشکلات و ارتباط بصری مفاهیم پیچیده به کار رود، اما با حجم کمتر و بیشتر به عنوان ابزاری برای پشتیبانی از گفتگوی رو در رو استفاده می شود، تا اینکه به عنوان یک اسناد نهایی در نظر گرفته شود.

توسعه نرم افزار در طول زمان تکامل یافته و همچنان در حال تغییر است. اگرچه مدل سازی و مستندسازی دقیق در متدولوژی های نسل سوم نقش بزرگی داشتند، اما با پیشرفت رویکردهای Agile و تمرکز بر انعطاف پذیری و سرعت، جایگاه این تکنیک ها تغییر کرده است. در حالی که رویکردهای Agile مدل سازی را به کلی رد نمی کنند، آن ها استفاده از مدل ها را به شیوه ای متفاوت توصیه می کنند، به گونه ای که پشتیبانی کننده ارتباطات فعال و سازنده باشند و نه به عنوان مستندات سنگین و دائمی.

سوال ۶

تفاوت «مدل ایجاد نرم افزار» مانند آبشاری یا حلزونی با «متدولوژی ایجاد نرم افزار» مانند XP یا RUP در چیست؟ انجمن علمی دانشکده مهندسی کامپیوتر خواستار «مدلی» برای برگزاری رویدادهای دانشجویی است. در طراحی این مدل، باید به ویژگی های زیر توجه شود:

- حق الزحمه ای به نیروهای برگزارکننده پرداخت نمی شود.
- احتمال عدم انجام وظایف توسط برگزارکنندگان به دلیل عدم تعهد رسمی.
- دانشجویان وقت محدودی دارند.
- موضوعات رویداد حول مباحث رشته ای مهندسی کامپیوتر است.
- هدف اصلی، یادگیری و سپس لذت بردن از کار تیمی است.
- مخاطبین عمدتاً دانشجویان و دانش آموزان هستند.

موارد مورد توجه در طراحی

- جامعه مخاطبین
- ثبت نام مخاطبین
- جذب داوطلبین برگزاری
- انتخاب افراد داوطلب
- تخمین هزینه ها

- حامی مالی
- تبلیغات و برندینگ
- خط زمانی رویداد
- هماهنگی‌های اداری

با توجه به مدلی که در قسمت قبل تهیه کرده‌اید، متدولوژی‌ای برای برگزاری یک رویداد خاص طراحی کنید. این متدولوژی باید موقعیت خاصی را در نظر بگیرد و به صورت دقیق به ویژگی‌های آن بپردازد.

جواب سوال ۶

تفاوت بین مدل ایجاد نرم‌افزار و متدولوژی ایجاد نرم‌افزار به نحوه‌ی دستورالعمل‌ها، فرآیندها، تکنیک‌ها و ابزارهایی برمی‌گردد که در هر کدام استفاده می‌شوند. بیایید این دو را با یکدیگر مقایسه کنیم:

مدل ایجاد نرم‌افزار:

مدل ایجاد نرم‌افزار به الگوهای کلی مراحل و فعالیت‌های لازم برای توسعه نرم‌افزار اشاره دارد. این مدل‌ها معمولاً رویکردی سطح بالا به فرآیند توسعه نرم‌افزار دارند و می‌توانند مفاهیم مختلفی را در بر بگیرند که تیم‌ها باید دنبال کنند.

- **آبشاری (Waterfall):** یک مدل خطی و ترتیبی است که در آن هر مرحله باید کاملاً تمام شود قبل از اینکه مرحله بعدی شروع شود. مثال: ابتدا تحلیل نیازمندی‌ها، سپس طراحی سیستم، پس از آن پیاده‌سازی، تست و نهایتاً نگهداری.
- **حلزونی (Spiral):** مدل حلزونی نیز مراحل آبشاری را دنبال می‌کند، اما با یک رویکرد تکراری که اجازه می‌دهد بازگشت به مراحل قبلی برای بهبود و اصلاح وجود داشته باشد. در هر دور، یک نسخه جدید و بهبود یافته از نرم‌افزار ساخته می‌شود.

متدولوژی ایجاد نرم‌افزار:

متدولوژی ایجاد نرم‌افزار نه تنها مراحل کلی فرآیند توسعه را تعریف می‌کند، بلکه تکنیک‌ها، ابزارها، و دستورالعمل‌های دقیقی را برای هر مرحله ارائه می‌دهد. متدولوژی‌ها معمولاً بسیار جامع‌تر هستند و می‌توانند شامل توصیه‌هایی برای برنامه‌ریزی، تخمین زمان، مدیریت پروژه، توسعه و نگهداری باشند.

- **XP (eXtreme Programming):** یک متدولوژی چابک است که بر توسعه تکراری، برنامه‌ریزی مداوم، و بهبود مستمر تأکید دارد. همچنین، این متدولوژی بر توسعه به شیوه‌ی جفتی، تست محور و داشتن بازخورد مداوم از مشتری تأکید می‌کند.
- **RUP (Rational Unified Process):** این متدولوژی یک فرآیند تکراری و افزایشی است که به تیم‌ها کمک می‌کند تا معماری نرم‌افزار را به خوبی تعریف کنند و مدیریت ریسک را در فرآیند توسعه ادغام کنند. RUP مجموعه‌ای از بهترین شیوه‌ها را در تمام جنبه‌های توسعه نرم‌افزار معرفی می‌کند.

بنابراین با توجه به تعریف‌هایی که داشتیم، تفاوت عمده در این است که مدل‌های توسعه نرم‌افزار بیشتر به الگوی کلی و توالی فعالیت‌ها توجه دارند، در حالی که متدولوژی‌های توسعه نرم‌افزار جزئیات دقیق‌تری از نحوه اجرای هر مرحله و اصول راهنما را ارائه می‌دهند و اغلب شامل راهنمایی‌های عملی‌تر و مشخص‌تر برای تیم‌های توسعه می‌شوند.

- **تمرکز بر فرآیند:** مدل‌های ایجاد نرم‌افزار بیشتر روی فرآیند توسعه متمرکز هستند. آنها مراحل و توالی عمومی فعالیت‌های مورد نیاز برای تولید نرم‌افزار را تعریف می‌کنند.
- **جامعیت پایین‌تر:** مدل‌هایی مانند آبشاری یا حلزونی معمولاً دستورالعمل‌های مشخص و جزئی برای پیاده‌سازی فرآیندها ارائه نمی‌دهند. آن‌ها چارچوب‌های کلی هستند که نحوه به دنبال کردن هر مرحله را به تیم‌های توسعه واگذار می‌کنند.
- **انعطاف‌پذیری کمتر:** مدل‌ها مانند آبشاری سفت و سخت‌تر هستند و تغییرات را در میانه‌ی پروژه به خوبی تحمل نمی‌کنند.
- **پیش‌بینی‌پذیری:** این مدل‌ها به دلیل ترتیب مشخص شده‌شان پیش‌بینی‌پذیری بیشتری در مراحل توسعه فراهم می‌آورند، که می‌تواند برای مدیریت پروژه مفید باشد.

بررسی دقیق‌تر متدولوژی‌های ایجاد نرم‌افزار (مانند XP یا RUP):

- **تمرکز بر جزئیات:** متدولوژی‌ها جزئیات دقیق‌تری از نحوه اجرای هر مرحله از فرآیند توسعه را فراهم می‌آورند، شامل روش‌ها، ابزارها، و دستورالعمل‌های خاص.
- **جامعیت بالاتر:** متدولوژی‌ها مجموعه‌ای از بهترین شیوه‌ها، قالب‌ها و استانداردهای صنعتی را ادغام می‌کنند که می‌تواند شامل توصیه‌های متعدد برای تمام جنبه‌های توسعه نرم‌افزار باشد.
- **انعطاف‌پذیری بیشتر:** متدولوژی‌ها مانند XP طراحی شده‌اند تا به تیم‌ها اجازه دهند به صورت چابک و با قابلیت پاسخگویی بالا به تغییرات پاسخ دهند.
- **تاکید بر بهبود مداوم:** متدولوژی‌ها اغلب شامل مکانیزم‌هایی برای بازنگری و بهبود مداوم فرآیندها هستند.

مثال:

- مدل آبشاری به شما می‌گوید که ابتدا نیامندی‌ها را جمع‌آوری کنید، سپس طراحی کنید، پس از آن کدنویسی، سپس تست و در نهایت به تحویل محصول بپردازید. این یک توالی خطی و غیرقابل بازگشت است.
- متدولوژی RUP، که یک متدولوژی تکراری و تدریجی است، به شما می‌گوید که چگونه باید نیامندی‌ها را با استفاده از تکنیک‌های خاص جمع‌آوری کنید، چطور باید معماری را مدل‌سازی کنید، چگونه ریسک‌ها را مدیریت کنید و چطور فرآیندهای کدنویسی و تست را به صورت تکراری و با ادغام تغییرات انجام دهید.

ارائه‌ی مدل برای برگزاری رویدادهای انجمن علمی

همان‌طور که می‌دانیم، 4MAT Learning Model مدلی برای طراحی تجربیات یادگیری و تدریس است که توسط دکتر برنیس مک‌کارتی توسعه یافته است. این مدل مبتنی بر این است که یادگیری در چهار فاز اصلی رخ می‌دهد: وابستگی (Why)، تفکر (What)، عملی (How)، و ابداع (If). با این حال، می‌توانیم از اصول این مدل برای طراحی و پیاده‌سازی مدل برگزاری رویدادهای دانشجویی استفاده کنیم. حالا در این مرحله، ما به تعریف دقیق هر مرحله از مدل خودمون می‌پردازیم و بررسی می‌کنیم که در هر مرحله چه کارهایی باید انجام شود و سپس تسک‌های تعریف شده در هر رویداد انجمن علمی را، اساین می‌کنیم به مراحل با توجه به تعاریفشان.

وابستگی (چرا؟ - Why) :

هدف‌گذاری و انگیزه: در این مرحله، لازم است که انگیزه‌های برگزاری رویداد را مشخص کرده و به اعضا و داوطلبان بفهمانیم که چرا مشارکت آن‌ها اهمیت دارد. این کار با ارائه مزایای شرکت در رویدادها، مانند یادگیری و تجربه کار تیمی، انجام می‌شود. به‌خصوص باید در فرم جذب استف‌ها و صحبت‌های دبیر-نایب‌دبیر رویدادها با افراد خارج از رویداد، باید به مزایای شرکت در رویداد اشاره شود و همچنین برگزارکنندگان اصلی رویدادها باید تلاش کنند تا مزایای زیادی ایجاد کنند برای حضور در تیم برگزاری رویداد و تمرکزشان را روی یادگیری مهارت‌های نرم و سخت قرار دهند.

تفکر (چه چیزی؟ - What) :

اطلاعات و داده‌ها: ارائه اطلاعات کلیدی در مورد رویداد، موضوعات، مخاطبین هدف، و فرآیند برگزاری به داوطلبان و شرکت‌کنندگان. این شامل توزیع دستورالعمل‌های دقیق، برنامه‌ها و مواد آموزشی است. در واقع، در این مرحله اطلاعات کلی و جزئی مربوط به نحوه شرکت در رویداد، نحوه استف شدن در رویداد، جزئیات شیوه برگزاری رویداد، مکان برگزاری‌ها، اطلاعات مربوط به شبکه‌های اجتماعی و باقی موارد، به اطلاع مخاطبان می‌رسد. این روند از طریق سوشال مدیاهای رویداد و همچنین کلامی می‌تواند شکل بگیرد.

عملی (چگونه؟ - How) :

فرایند برگزاری: توسعه یک نقشه عملی برای برگزاری رویداد که شامل جذب داوطلبین، ثبت‌نام، تخمین هزینه‌ها، جذب حامی مالی، تبلیغات و برندینگ، خط زمانی رویداد و هماهنگی‌های اداری می‌شود. در این فاز باید دستورالعمل‌های عملی و واضحی برای هر یک از این بخش‌ها ارائه شود. این بخش خیلی دقیق به روند برگزاری رویداد می‌پردازد و نحوه برگزاری را مشخص می‌کند. در این‌جاست که دبیر-نایب‌دبیر رویدادها، درباره‌ی موارد مختلف مربوط به رویدادها تصمیم‌گیری می‌کنند و برنامه‌ی شیوه برگزاری را می‌ریزند.

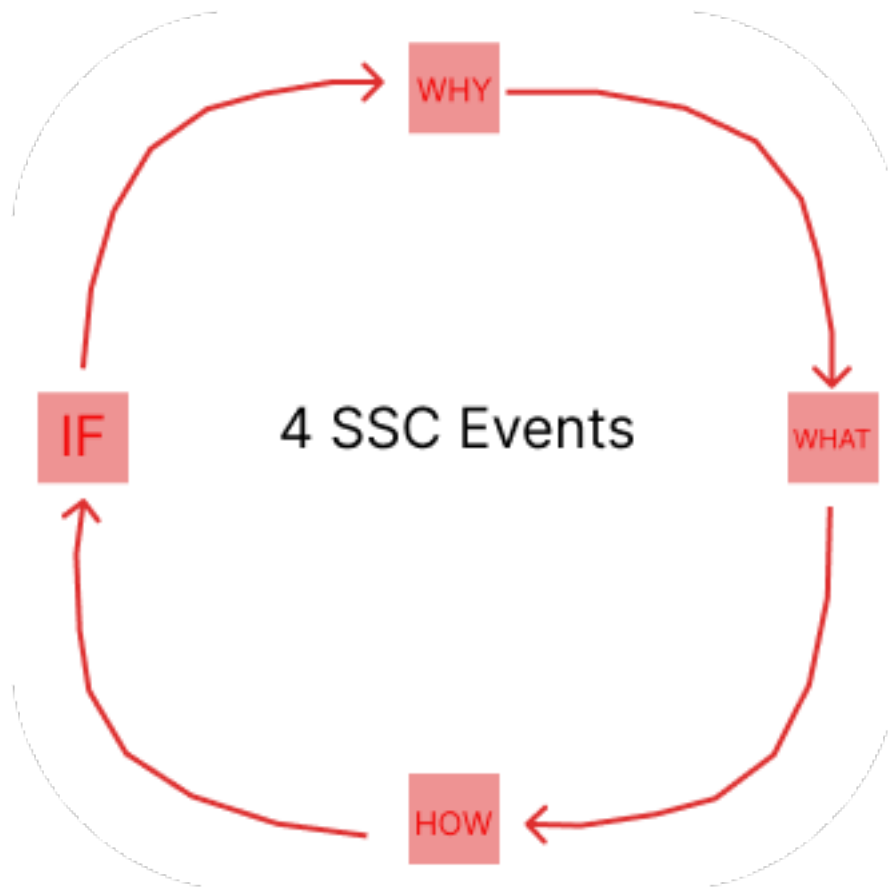
ابداع (اگر چه چیزی؟ - If) :

بازخورد و بهبود: در این مرحله، فرصتی برای ارزیابی و بازاندیشی فراهم می‌شود. پس از هر رویداد، تیم برگزاری باید دور هم جمع شوند و در مورد آنچه خوب پیش رفت و چه چیزهایی نیاز به بهبود دارند بحث کنند. این مرحله همچنین فرصتی برای بررسی احتمالات جدید و نوآوری‌های احتمالی در رویدادهای آتی است. جلسات بازبینی رویداد در این مرحله برگزار می‌شوند و مستندات در این مرحله به کار می‌آیند. در واقع تیم مستندسازی با هدف این مرحله تشکیل می‌شود در رویدادها.

در طراحی یک مدل برگزاری رویدادهای انجمن علمی، موارد زیر باید با دقت مورد توجه قرار گیرند تا اطمینان حاصل شود که هر جنبه از رویداد به درستی برنامه‌ریزی و اجرا می‌شود:

جامعه مخاطبین

- تعیین جامعه مخاطبین هدف با توجه به موضوع و هدف رویداد.
- شناخت دقیق نیازها و علایق مخاطبین برای طراحی محتوای مرتبط و جذاب.



ثبت نام مخاطبین

- ایجاد فرم های ثبت نام آسان برای استفاده که تمامی اطلاعات لازم را جمع آوری کند.
- طراحی سیستم ثبت نام آنلاین و اتوماسیون برای کاهش اشتباهات و تسهیل پروسه ثبت نام.

جذب داوطلبین برگزاری

- تعیین نقش ها و مسئولیت های داوطلبین و ایجاد فراخوان عمومی برای جذب داوطلبین.
- انتشار فراخوان در شبکه های اجتماعی، تابلوهای اعلانات، و دیگر پلتفرم ها برای رسیدن به جامعه گسترده تری از داوطلبین.

انتخاب افراد داوطلب

- برگزاری مصاحبه های کوتاه و ارزیابی های مهارتی برای انتخاب داوطلبین مناسب.
- تاکید بر تیم سازی و تعامل برای اطمینان از همکاری موثر داوطلبین.

تخمین هزینه ها

- تهیه یک جدول بودجه دقیق که تمام هزینه های مورد انتظار را پوشش دهد.
- بررسی منابع مالی موجود و تعیین استراتژی برای تامین کسری بودجه احتمالی.

حامی مالی

- شناسایی و جذب حامیان مالی با ارائه پروپوزال‌های قانع‌کننده و مزایای تبلیغاتی متقابل.
- مذاکره با حامیان بالقوه و ایجاد قراردادهای شفاف و متعهدانه.

تبلیغات و برندینگ

- طراحی یک کمپین تبلیغاتی جامع که شامل رسانه‌های دیجیتال، چاپی و شاید برودکست باشد.
- ایجاد یک هویت برند قوی و به یادماندنی برای رویداد که در تمامی مواد تبلیغاتی به کار گرفته شود.

خط زمانی رویداد

- تدوین یک برنامه زمانی دقیق برای تمام جنبه‌های رویداد، از پیش‌برنامه‌ریزی تا پیگیری پس از برگزاری.
- انتخاب تاریخ و زمان مناسب که با رویدادهای دیگر تداخل نداشته باشد و برای اکثر مخاطبین قابل دسترس باشد.

مکان برگزاری

- انتخاب مکانی مناسب با ظرفیت کافی و امکانات لازم برای اجرای رویداد.
- بررسی دسترسی به مکان، امنیت، پارکینگ و سایر عوامل لجستیکی.

برنامه‌ریزی محتوا

- تعیین اسپیکرها، موضوعات ارائه، کارگاه‌ها، و بحث‌های پنلی.
- طراحی برنامه‌ای جذاب و متنوع که مخاطبین را ترغیب به مشارکت فعال کند.

لوجستیک و تدارکات

- تدارک تجهیزات لازم مانند سیستم‌های صوتی/تصویری، وسایل ارتباطی، و نیازهای پذیرایی.
- ایجاد تیم لجستیکی برای مدیریت و نظارت بر جزئیات در طول رویداد.

ارزیابی و بازخورد

- طراحی نظرسنجی‌ها و فرم‌های بازخورد برای ارزیابی تجربه شرکت‌کنندگان.
- برنامه‌ریزی برای جمع‌آوری و تجزیه و تحلیل داده‌های به دست آمده به منظور بهبود رویدادهای آتی.

مدیریت بحران

- پیش‌بینی مشکلات احتمالی و تهیه برنامه‌های جایگزین برای حوادث ناگهانی.
- تعیین فرایندهای ارتباطی برای مواجهه با شرایط اضطراری.

با دقت به این موارد و انعطاف‌پذیری در برابر تغییرات احتمالی، یک انجمن علمی می‌تواند رویدادهای موفق و ماندگاری را برگزار کند که به ارتقای دانش و همکاری بین اعضا کمک می‌کند.

در بخش‌بندی بالاتر ذکر شده بود که هر کدوم از این موارد در کدوم یک از ۴ بخش مدل قرار می‌گیرند، برای همین در زیر هر کدام، فقط به توضیح روند کلی تعیین شده در مدل برای هر کدام، پرداختیم.

تعریف متدولوژی برگزاری یک رویداد خاص

تعریف موقعیت و ویژگی‌های رویداد

- نوع رویداد: وبینار تخصصی با پنل‌های بحث و کارگاه‌های آموزشی.
- هدف رویداد: افزایش آگاهی درباره چالش‌های امنیت سایبری و ارائه راهکارهای نوآورانه.
- مخاطبین هدف: متخصصان IT، دانشجویان حوزه تکنولوژی، شرکت‌های فعال در حوزه امنیت سایبری.
- زمان بندی: یک رویداد ۳ روزه در ماه دسامبر.
- پلتفرم: برگزاری آنلاین از طریق پلتفرمی مانند Zoom یا Webex با قابلیت‌های امنیتی بالا.

تحلیل نیازمندی‌ها و محدودیت‌های رویداد

- تکنولوژی مورد نیاز: نیاز به سرورهای قوی برای پشتیبانی از وبینار و پخش زنده.
- محتوای آموزشی: جذب سخنرانان مطرح، مدرسان مجرب و ارائه‌دهندگان محتوا.
- بودجه: تعیین بودجه متناسب با هزینه‌های تکنولوژیک و تبلیغات.
- زمان‌بندی: هماهنگی با تقویم‌های بین‌المللی تا با رویدادهای مشابه تداخل نداشته باشد.

برنامه‌ریزی جامع برای برگزاری

- برنامه رویداد تدوین جدول زمانی دقیق برای سخنرانی‌ها، پنل‌ها و کارگاه‌ها.
- تبلیغات طراحی کمپین‌های تبلیغاتی موثر در شبکه‌های اجتماعی و انجمن‌های تخصصی.
- حامیان مالی جذب حامیان مالی با ارائه بسته‌های تبلیغاتی اختصاصی.
- ثبت‌نام ایجاد سیستم ثبت‌نام آسان و امن برای شرکت‌کنندگان.

پیاده‌سازی و اجرای رویداد

- فنی: راه‌اندازی و آزمایش زیرساخت‌های فنی قبل از رویداد.
- مدیریت داوطلبین: تربیت و هماهنگی تیم پشتیبانی برای راهنمایی و پاسخگویی به شرکت‌کنندگان.
- پشتیبانی زنده: تأمین پشتیبانی فنی به صورت لحظه‌ای در طول برگزاری رویداد.

ارزیابی و بازبینی پس از اتمام رویداد

- نظرسنجی‌ها: اجرای نظرسنجی‌ها فوراً پس از پایان هر بخش و در پایان رویداد.
- تحلیل داده‌ها: بررسی داده‌های جمع‌آوری شده برای فهمیدن نقاط قوت و ضعف رویداد.
- گزارش‌دهی: تهیه گزارش کامل از رویداد و به اشتراک‌گذاری با سخنرانان و شرکت‌کنندگان.
- طرح بهبود: توسعه طرح‌های بهبود برای اجرای موفق‌تر رویدادهای بعدی.

توسعه محتوا و مواد آموزشی

- تهیه محتوا: همکاری با سخنرانان برای تهیه اسلایدها، ویدئوها و مواد دوره‌های آموزشی.
- دسترسی به مواد: فراهم کردن دسترسی به محتوا برای شرکت‌کنندگان قبل و بعد از رویداد.
- تنوع بخشی: اطمینان حاصل کردن از تنوع بخشی به مواد آموزشی برای پوشش دادن به انواع یادگیری.

برقراری ارتباط و شبکه‌سازی

- فضاهای تعاملی: ایجاد فضاهای تعاملی مجازی برای ارتباط بین شرکت‌کنندگان و سخنرانان.
- فعالیتهای شبکه‌سازی: برنامه‌ریزی برای فعالیتهای شبکه‌سازی مانند جلسات Q&A و میزگردها.
- استفاده از رسانه‌ها: تشویق شرکت‌کنندگان به استفاده از هشتگ‌های رویداد در شبکه‌های اجتماعی.

مدیریت ریسک و مسائل امنیتی

- برنامه‌ریزی برای امنیت: تضمین امنیت سایبری و حفظ حریم خصوصی در طول برگزاری وبینار.
- مدیریت ریسک: شناسایی و ارزیابی ریسک‌های احتمالی و تدوین برنامه‌های مدیریت بحران.

تبلیغات و ارتقاء رویداد

- تبلیغات پیش از رویداد: اجرای کمپین‌های هدفمند برای جذب حداکثری شرکت‌کنندگان.
- مشارکت‌های استراتژیک: همکاری با انجمن‌ها و سازمان‌های مرتبط برای افزایش دید و اعتبار.
- تبلیغات درون برنامه‌ای: استفاده از فرصت‌های درون رویداد برای ترویج برنامه‌های آتی.

پایان‌بخش و تحویل محتوا

- ارائه گواهی‌نامه‌ها: صدور گواهی شرکت برای شرکت‌کنندگان و سخنرانان.
- بایگانی و دسترسی: ارائه دسترسی به ضبط جلسات برای بازبینی در آینده.
- پیگیری پس از رویداد: ارسال ایمیل‌های تشکر و دعوت برای فیدبک به شرکت‌کنندگان.

نتیجه‌گیری

با دنبال کردن این متدولوژی، می‌توان یک رویداد مجازی موفق برگزار کرد که نه تنها در زمان برگزاری، بلکه قبل و بعد از آن نیز تاثیر مثبت و دوامی درازمدت بر جامعه هدف خود داشته باشد.
