User inter Face

Design

طراحی لایه واسط کاربر

موفقیت ما در طول ساختن برنامه های کاربردی بستگی به توانایی ما در پیش بینی اجرای برنامه ها دارد . برخی از جنبه های اجرای برنامه تقریبا برای پیش بینی آسان می باشد . مثلا به راحتی می توانیم ظرفیت برنامه را در ترم های بخش های داده ای یا عناصر تصویر تخمین بزنیم . دیگر جنبه های اجرا مثل اندازه ی برنامه و سرعت پاسخ گویی یا قابلیت اطمینان برنامه برای تعیین کردن سخت است . اغلب موضوعات عمده ای هستند که به طور بهینه در طول برنامه تخمین زده شده اند تاثیر بهینه سازی در این تخمین ها در برنامه هنگام کامل شدن در مواجه با نیازهای اساسی با شکست روبرو می شود . واسط کاربر تنها جزء قابل تعامل برنامه است که در اجرا قابل پیش بینی نمی باشد . مثلا آن جزء از برنامه که تصمیم می گیرد چگونه کاربر و کامپیوتر با هم ارتباط داشته باشند (و این قابل تاسف است که باید این گونه باشد) برای طراحی واسط کاربر تاثیر زیادی بر مقبولیت برنامه دارد . ناتوانی ما در پیش بینی اجزای واسط کاربر باعث می شود که کاربر در راه ها ی غیر منتظره واکنش نشان دهند در جاییکه آنان برای اولین بار از برنامه استفاده می کنند بیشترین شگفتی زمانی اتفاق می افتد

که برنامه نویس با اولین کاربر و کاربر مبتدی همراه شود تا اجرای برنامه را توضیح دهد

:

بدین صورت

برنامه نویس : حالا که شما این مدار را کشیده اید ممکن است بخوا هید چنـد راه آن را تغییر دهید .

کاربر : بله پس باید یک عنصر را حذف کنیم . چه طور باید این کار را انجام دهـیم ؟

برنامه نویس : روی آیتمی که برچسب CD دارد کلیک کنید .

کاربر : CD ؟

برنامه نویس : این برای حذف کردن یک عنصر است .

کاربر : بله . خوب اجرا می کنیم راستی چه اتفاقی می افتد ؟

برنامه نویس : شما در مود تحلیل هستید باید AM را به جای CD انتخاب کنید .

کاربر : چه جالب من در حال اشاره کردن به CDهستم . چگونـه مـی تـوانیم از مـود تجزیه بیرون بیاییم ؟

برنامه نویس : کافی است control-Q را تایپ کنید .

کاربر : (Type C-O-N-T-R)

برنامه نویس : نه ، کلید کنترل (Kontrol-Key) را فشار بده و بعد دکمه ی Q را بزن .

کاربر : متاسفم اشتباه کردم بله من باید دوباره شروع کنم .

برنامه نویس : حالا به عنصر برای پاک کردن اینها ؟

کاربر : بله هیچ اتفاقی رخ نداد . آیا من اشتباهی انجام داده ام ؟

برنامه نویس : نه تو هیچ اشتباهی مرتکب نشدی : تو عناصر را پاک کردی اما هنوز برنامه این را از صفحه ی نمایش Remove نکرده است .

کاربر : پس کی Remove خواهد شد ؟

برنامه نویس : هنگامی که تو Control-j را برای دوباره کشیدن تصویر تایپ کنی .

کاربر : من این کار را انجام خواهم داد ما این هستیم ولی تنها یک قسمت Component ها ، Remove شده است .

برنامه نویس : ببخشید من فراموش کردم . شما باید نصفی از این عناصر را به طور جداگانه پاک کنید . و تنها دوباره به CD اشاره کنید .

کاربر : خیلی خوب حالا چه اتفاقی می افتد ؟

برنامه نویس : تو الان دوباره در مود تجزیه هستی Control-Q را تایپ کن .

کاربر : کنترل پس Q کجاست ؟ اینجاست چرا همه جا خالی شد و هیچ اثری نمایش داده نشده است .

برنامه نویس : شما Q را تایپ کردید . نه Control-Q ، بنابر این برنامه سیستم عملیاتی رها می شود . من واقعا متاسفم ، اما ما خیلی چیزها را از دست دادیم و باید دوباره از اول شروع کنیم .

کاربر : بله چقدر بد ما می توانیم تا هفته ی دیگر این کار را عقب بیندازیم ؟

آنچه که تا این جا دیدید نمونه ای از یک لایه واسط کاربر طراحی شده ناچیز بود و تاثیر آن که می تواند یک برنامه ی تعاملی سودمند داشته باشد .

کاربر بدشانس مجبور است تلاش کند که دستوراتی که خارج از کنترل هستند مانند C D و Control-Q را به خاطر بسپارد .

برنامه اغلب به دستورات داده شده جواب نمی دهد وقتی که جواب می دهد کاربر اغلب به خاطر نتیجه متعجب یا گیج می شود .

و همچنین این اشتباهات کوچک در برنامه هستند که باعث واکنش های غافلگیر کننده توسط برنامه می شود .

این امر خیلی مهم است که به طراحی واسط کاربر تعاملی ، توجه ویژه نشان دهیم . نه تنها واسط کاربر کم تجربه و ضعیف در امور کلیدی برای یادگیری سخت است بلکه باعث می شود برنامه برای اجرا کردن برای استفاده ی یک کاربر با تجربه بی مصرف باشد .

در مورد های مهم که مثال بالا نمونه ای از آن بود تمرین کل اجرا ممکن است در اثر طراحی ضعیف غیر موجود شود و ممکن است این امر ثابت گردد که یاد دادن برنامه به کاربر موردنظر با این طراحی ضعیف غیر ممکن است و یا اینکه واسط کاربر ممکن است آنقدر غیر قابل مصرف و غیر قابل اعتماد باشد که هزینه ی استفاده از برنامه تامین نشود .

28-1 COMPONENTS OF THE USER INTERFACE

28-1 : عناصر واسط کاربر .

واسط کاربر به طور طبیعی به ۴ قسمت تقسیم می شود یکی از این ها زیر لایه ۳ تای دیگر است که این همان مدل کاربر می باشد .

مدل تصویری که توسط کاربر اطلاعات ، فرم داده می شود و فرایند هایی که برای این اطلاعات به کار برده است و بدون این امر ، مدل کاربر فقط می تواند یکسری اعمال

کورکورانه را دنبال کند مانند یک آشپز غیر حرفه ای که دستورالعمل را دنبال می کند

.

مدل ، کاربر را قادر می سازد تا مفهوم گسترده ای از آنچه که برنامه می خواهد انجام دهد را توسعه دهد حتی اگر از دانش کامپیوتر چیزی نداند یا خیلی کم بداند . با کمک مدل ، کاربر می تواند استراتژی خودش را ، برای برنامه های عملیاتی شکل دهد .

به عنوان مثال : در طراحی سیستم راهنما خلبان هواپیما .

در ابتدا کاربر مدل را درک کرده است . و همچنان کاربر به فرامینی جهت اداره کردن مدل احتیاج دارد . سیستمی از فرامین که می توانیم استفاده کنیم دستورزبان نام دارد و به عناصر ثانویه واسط کاربر شکل می دهد .

(البته بیشتر از شما با این دستورات آشنا هستید .) یا مانند یکسری ماشین های محاسباتی که با یکسری فرامین و دستورات خاص خودشان کار می کنند .

مثلا یک سیستم ماشین حساب یا ماشین تحریر را در نظر بگیرید . در این ماشین ها از یکسری فرامین کلی استفاده می شود که ما می توانیم با فهمیدن آن دستورات خیلی راحت طرز استفاده از این ماشین ها را فرا بگیریم .

به طور ایده ال برنامه های کامپیوتر باید دستورزبان های مساوی و به طور طبیعی داشته باشد .

عنصر سوم از واسط کاربر (فیدبک ها) Feed Back هستند که کاربرها را در اجـرای برنامه در کامپیوتر کمک می کند .

Feed Back ها در شکل های مختلف وجود دارند : تصدیق اعلام وصول دستورات یـا تصدیق درستی زبان ـ پیام های متنی و توضیح دار و غیره و

بعضی از فرم های Feed Back اساسا برای کمک به تجارب کاربر ها تهیه شده انـد از طرف دیگر بعضی از دستورزبان ها در برنامه بـه طـور ذاتـی وابسـته بـه Feed Back هستند .

فیدبک به کاربر کمک می کند بـرای اینکـه وی مطمـئن شـود کـه دسـتوراتش دقیقـا دریافت شده است و کاملا برای کامپیوتر بدیهی و واضح می باشد . و کاربر تاثیر واقعـی را روی برنامه مشاهده نماید .

چهارمین عنصر نمایش اطلاعات اسـت . کـه ایـن امـر بـرای نشـان دادن حالـت هـای اطلاعات برای کاربر ضروری مسی باشد. عامل تصویر می تواند نـوعی تصـدیق و تاییـد برای کاربر باشد به این که مدل کاربر درست است یا نه در جایی کـه مـدل وابسـته بـه واقعیت و واقع گرایی می باشد صدق می کند و همچنانکه مـا بایـد بـه واقـع گرایـی در تصاویر نمایشی خود بیشتر توجه داشته باشیم و سعی کنیم تصـاویر طراحـی شـده بـه واقعیت نزدیک تر باشند و سعی کنیم جامع تر شوند .

در جایی که چند مدل با هم ترکیب می شوند و برای طراحی انتخاب می شود این امر صدق می کند .

ما باید بوسیله ی یک عمل انتخابی مناسب در حد وسط و شبیه سازی گرافیکی در تقویت و بهبود این مدل ها کوشش کنیم .

این بخش فرعی در این اجزا در طراحی واسط کاربر مهم است زیرا این عامل ما را به دسته بندی کردن در مشکلات بوجود آمده در طراحی قادر می سازد و همچنین کار طراحی واسط کاربر را به خرده کارهای متناظر کوچک تفکیک می کند برای مثال هنگامی که ما یک دستور زبان را طراحی می کنیم ما باید بدانیم که Feed Back ، فیدبک هر دستور باید تهیه شود و این (فیدبک ها) Feed Back های تکنیکی برای نمایش اطلاعات باید از خروجی های مشابه استفاده کنند . هنگامی که ما این عناصر را طراحی می کنیم اغلب به مدل کاربر مراجعه می کنسم ، که تغییرات کوچک تری را بوجود بیاوریم برای منعکس کردن تغییرات در دستورزبان یا در نمایش اطلاعات . بدین سان ما باید مرکز توجه مان را از یک عنصر به عنصر دیگر تغییر دهیم . با این همه ما می توانیم استراتژی های جداگانه به کار بریم و قانون هایی را برای هر عنصر طراحی کنیم و مطمئن شویم که هر یک از ۴ عنصر نام برده به طور رضایت بخش عمل می کنند .

Task Analize

تجزیه کار پیروی کردن از قسمت کاوش کردن در جزئیات هر یک از چهار عنصر شرح داده شده :

مدل واسط کاربر ـ دستورزبان ـ فیدبک و نمایش اطلاعات .

البته پیامدهای دیگری در طراحی واسط کاربر وارد می شوند و باید در حافظه بوسیله ی طراح نگهداری شوند . برای مثال در فرایند طراحی باید یک تخمین واقعی زده شود آن هم از یکسری محاسبات و منابع اطلاعات که برای وی موجود می باشد و همچنین طراح باید مطمئن شود که طراحی واسط کاربرش هزینه ی سنگینی برای وی متحمل می شود یا نه و همه چیز در کل باید به جا باشد در همین موقع است که وی باید احتیاجات کاربر را در نظر بگیرد در این جا می توان به این نکته اشاره نمود که محرک اصلی کار طراحی همین عامل می باشد . پس طراحی باید در ابتدای کار با درایت و به طور واضح تمام خواسته ها را معین نماید . در این جا در می یابیم :

که در یک طراحی تجزیه کار (Task Analize) مهم است و باید زیرکانه انجام و پیاده سازی شود.

همچنین که احتیاجات کاربر در نظر گرفته می شود یک راه دیگر هم استفاده از تابع های موجود مورد نیاز این کار است در واقع تجزیه کار بسیار مفید است و بعدا در آینده

این موضوع را متوجه می شویم که این گونه طراحی برای استفاده u ser کار آمده بود . تجزیه کار در واقع اجرای کاربر را در نظر می گیرد و همچنین محیط کار آن ها را بررسی می کند و نتیجه ی این بررسی ها باید مورد تجزیه واقع شود .

یک گزارش نوشته شده از تجزیه کار اغلب با یک نکته آغازین برای طراحی واسط کاربر شکل می گیرد .

تجزیه کار اساسا خودش یک مهارت قابل توجه است که مورد احتیاج فرایند ها می باشد .

و همچنین می تواند یک تجربه به حساب بیاید . اغلب تجربه کار به دوش متخصصانی است که یک گزارش را تولید میکنند و قسمت هایی از وظایف طراحی به دوش آنان می باشد . در پایان این فصل یکسری مثال وجود دارد که تعاریفی کوتاه و مختصر از احتیاجات کاربر را به ما می دهد .

مدل کاربر

هنگامی که ما یک ذهنیت از قبل تعیین شده داریم کاربر می تواند برای به کار گیری برنامه های تعاملی بر اساس استفاده از دستورالعمل های همراه و مورد نیاز آنان تربیت شوند ، قوانین و راهنمای کاربر اغلب شامل دستورالعمل های کوچک همانند مثال آورده شده زیر می باشند .

بوجود آوردن فهرست هفتگی .

۱- تایپ کن EDITMUNUTXT و کلید بازگشت (RETURN) را فشار بده .

۲- هنگامی که منوی اصلی روی صفحه نمایش ظاهر شد مکان نما را روی کلمه ی mondoy-dish فشار بده و دکمه ی وسط موس را یک بار فشار بده . کلمه باید دارای underline (زیر خط) شود .

۳- Rkey را فشار بده .

۴- توضیحات روزانه را در لیست تایپ کن . این کار را با فشار دادن ESC دنبال کن .

۵- مرحله ی ۲ و ۳ را تکرار کن و این مراحل را برای هر کدام از روزها تکرار کن .

۶- هنگامی که تمامی لیست های ۵ روز تایپ شده کلید H را بعد از ESC فشار بده و نتیجه ی چاپی بدست آمده از لیست را پرینت کن .

۷- Q و ESC را تایپ کن.

البته دستورالعمل مورد نظر برای استفاده برنامه های تعاملی چند تا نقطه ضعف دارد .

یکی از جدی ترین نقاط ضعف این است که کاربر را از درک قابل توسعه از برنامه که وی در حال اجرا کردن است دلسرد می کند و از آن یک وجه غیر جذاب می سازد و

ممکن است مثلا در هنگامی باشد که کاربر می خواهد بداند که چه کاری باید انجام دهد اگر دستورالعمل ها کار نکند.

برای مثال : برای یک مقاله ممکن است در ۶ قدم در یک چاپگر گنجانده شود .

ـ آیا کاربر باید به مرحله ی اول بازگردد و تمام فرایند و مراحل را کامل بگذراند ؟ یا حقیقتا او می تواند مرحله ۶ را تکرار کند بعد از آزادسازی چاپگر ؟

ـ او چه کاری باید انجام دهد اگر یک کلمه ی غلط در مرحله ۲ وجود داشته باشد .

ممکن است برای ما پاسخ دادن به این سوال بدیهی و واضح باشد اما برای کاربر آن لیست دربردارنده ی مفهوم اساسی برای ساختن یک جریان از عملیات معقول تر از بقیه نمی باشد .

برای حفظ کردن کاربر در برابر این قبیل مشکلات ما بایستی به وی از طریق فهماندن مفاهیم ادراکی کمک نمائیم . در بسیاری از موارد یک مدل کاربر می تواند بر روی یک مدل خیلی آسان با یک مفهوم غیر تکنیکی هم پایه گذاری شود .

برای مثال ما می توانیم

- کاربری را از دستور العملهای شرح داده شده استفاده می کرد را اعمال کمکی زیر آماده کنیم

۱- ویرایش برنامه به کاربر این اجازه را می دهد که بتواند یک فایل مبتنی را ویرایش کند و روی آن فعالیت کند برای شروع عملیات ویرایش Edit را تایپ می کنیم که بوسیله نام فایل دنبال می شود و توسط RETurn ختم می شود.

۲- هر کلمه ای در متن ممکن است توسط اشاره گر با فشار بین وسط موس انتخاب شده و underline گردد.

۳- در واقع انتخاب شدن متن را تغییر نمی دهد. برنامه یکسری از فرامین را دارد که ممکن است هیچ وقت انتخاب نشود برای دستور Replace، R را تایپ می کنیم برای دستاوری HARDCOPY (H)و (Q)Quit را تایپ می کنیم 3 . برای کامل کردن و تمام کردن هر کدام از دستورات و خارج شدن ESC را می زنیم

۴- نرمال جایگزینی کلمات را با متن تایپ شده قبل از اینکه ESC را فشار دهیم جایگزین می شود. شاید اینها کافی نباشد به این که کاربر تمام مشکلاتش حل شود اما به هر حال کاربر را با مدل کاربری که در جای خودش دارای ارزش است برای فهمیدن منظور هر کدام از مراحل در منوها آماده کند.در عوض ما باید مدل کاربر را از ابتدای کار ملاحظه کنیم. در امر طراحی مدل کاربر می تواند مفید باشد برای رسیدن به ۲ نکته!

۱- مدل کاربر یک مدل هوشمند است و یک مدلی به عنوان چهار چوبی است برای توسعه استراتژی هایی که در انجام اجرا برنامه وجود دارند و قابل قیاس و تشابه با گرامر زبان های خارجی می باشد .

۲- مدل کاربر باید مفاهیم آشنا را در بر گیرد و بکار ببرد و برنامه باید پذیرش آن را داشته باشد در کل باید مفاهیمی را به کار ببریم که برای درک کاربر آسان باشد اگر او یک مهندس الکترونیک است ما باید به او به عنوان یک عنصر حافظه – ورودی – سیم های انتقال و obj هایی از این قبیل بدهیم.

۳- اگر کاربر یک آرشیتک (معمار) باشد ما باید به او این اجازه را بدهیم برای اینکه درها و پنجره ها و دیواره ها را دستکاری کند.

۴- مفاهیم آشنا برای استفاده مدل کاربر را قابل درک تر و مستقیم تر برای کاربر جلوه می دهد.

برای مثال ما می توانیم

- کاربری از دستور العملهای شرح داده شده استفاده می کرد را اعمال کمکی زیر آماده کنیم.

۱. ویرایش برنامه به کاربر این اجازه را می دهد که بتواند یک فایل متنی را ویرایش کند و روی آن فعالیت کند. برای شروع عملیات ویرایش EDIT را تایپ می کنیم که بوسیله نام فایل دنبال می شود و توسط Return ختم می شود.

۲. هر کلمه ای در متن ممکن است توسط اشاره گر با فشار بین وسط موس انتخاب شده و underlineگردد .

در واقع انتخاب شدن را تغییر نمی دهد.برنامه یکسری از فرامین را دارد که ممکن است

برای دستور Replace ،

هیچ وقت انتخاب نشود. برای دستور ، را تایپ می کنیم. برای دستور را تایپ می کنیم

۳. برای کامل کردن و تمام کردن هر کدام از دستورات و خارج شدن ESC را می زنیم .

۴. فرمان جایگزینی کلمات را با متن تایپ شده قبل از اینکه ESC را فشار دهیم جایگزین می شود .

شاید اینها کافی نباشد به این که کاربر تمام مشکلاتش حل شود اما به هر حال کاربر را با مدل کاربری که در جای خودش دارای ارزش است برای فهمیدن منظور هر

کدام از مراحل در منوها آماده کند در عوض ما باید مدل کاربر را از ابتدای کار ملاحظه کنیم . در امر طراحی مدل کاربر می تواند مفید باشد .

برای رسیدن به دو نکته

۱-مدل کاربر یک مدل هوشمند است و یک مدلی بعنوان چهار چوبی است برای توسعه استراتژ هایی که در انجام اجرا ی برنامه وجود دارد و قابل قیاس و متشابهاً گرامر زبانهای خارجی می باشد.

۲-مدل کاربر نباید مفاهیم آشنا را در بر بگیرد و بکار ببرد در برنامه باید پذیرش آنها را داشته باشد در کل باید مفاهیمی را بکار ببریم که برای درک کاربر آسان باشد ، اگر او یک مهندس الکترونیک است ما باید به او عنصر حافظه ای ورودی سیمهای انتقال و OBJ از این قبیل بدهیم .

اگر کاربر یک آرشیتکت (معمار) باشد ما باید به او ایت اجازه را بدهیم برا یاینکه دربها و پنجره ها و دیواره دستکاری می کند .

مفاهیم آشنا برای استفاده مدل کاربر را قابل درک تر و مستقیمتر برای کاربر جلوه می دهد .

و همچنین مفاهیم ساده وآشنا بسیار رحتتر است موارد یادگیری و ما را برای یک شروع خوب برای طراحی یک مدل کاربر آماده میکند . وهمچنین به این وسیله می توانیم

به تصحیح کردن و ادامه دادن مدل مورد نظر اقدام کنیم و برای کمک بکار که خودش را با برنامه و دستوراتش سازگار کند آسان می باشد . این موضوع بسیار مهم است گرچه نمی تواند بصورت لفظی در چاپ کردن OBJ های عامیانه خودمانی استفاده شود .

بسیاری از محیطها برای واکنش کامل یک برنامه کامپیوتری پیچیده و غیر بدیهی می باشد . در حالیکه تفاوت قابل قیاسی بین دنیای واقعی و مجازی وجود دارد و ممکن است درک این تفاوت ها برای کاربر سخت باشد پس ما می توانیم شبیه سازی انجام دهیم مثل شبیه سازی یک ماشین تحریر خیلی دقیق روی یک صفحه نمایش .

اما اگر ما از خدمان تغییر ایجاد کنیم و یک چیز تازه ای به وجد بیاوریم در ساختن همچین چیزی مثلاً کلید Back space کاراکترها را Delete می کند ولی ممکن است ما با دادن این دستور برای یک تایپیست که با تجربه که همیشه از Back spacing برای دستور Underline دادن به کلمه استفاده میکند ، مشکل ایجاد کنیم و این ممکن برای طراحی یک مدل که تا حدی با کاشی تایپ تفاوت دارد بهتر باشد .

بنا براین برای اجتناب از این نکات گیج کننده ما باید بطور صحیح و شناخته شده عمل کنیم .

بطور کلی این مسئله بسیار مهم است هنگامی که در حال طراحی مدل کاربر هستیم و می توانیم به مشکلاتی که توسط سخت افزار و نرم افزار بوجود می اید رسیدگی کنیم .

عناصر دیگر واسطه کاربر مستیماً بر روی این مشکلات تأثیر می گذارد .

موضوعات و حرکات

یک کراه سود مند نبرای نمایش مدل کاربر برای اهداف گفتگو ، مستند سازی ، یک سری Obj می باشد . هر Obj یک بخش از اطلاعات می باشد که هر کاربر بر روی آن کنترل دارد و قادر باشد آنرا نمایش دهد .

و حتی در مورد آن سئوال کند و یا مکی تواند آنرا تغییر دهد یا قادر است آنرا خراب کند و یا Obj دیگری در همان مکان ایجاد کند .

اغلب یک واسطه سلسله مراتبی بین Obj موجود می باشد برای مثال در جایی که یک ویرایشگر متن ساده ترین Obj یک کاراکتر تنها ، مجموعه ای از کاراکترهای حروف ، شکل خطای پاراگراف حروف شکا صفحات یک سند کامل می باشد . اعمال کارهایی هستند که کاربر ها می توانند بر روی Obj بکار برند هر عمل می تواند بر روی هر Obj انجام شود .

ویرایشگر متن غالباً به ما توضیح اجازه دهد که کاراکترها را کارها ، خطاها ، پاراگراف و صفحات و همه سندی را چاپ می کند . علاوه بر آن ما می توانیم هر کاری که انجام می دهم فقط به یکسری از Obj اعمال نمائیم همانگونه که در صفحات بعد مشاهده می نمائیم مکاتبات نزدیک بین مدل کاربر و دستورات دیگر وجود دارد که زبان دستورات را مشکل می هد این مکاتبات تک به تک نیستند بلکه ما می توانیم دستوراتی بیشتری برای انجام یکسری از اعمال اضافه نمائیم و یا ما اجازه انجام دستورات کتبی با تعیین کننده مناسب که کارهای مختلفی را انجام دهد داریم .

مجموعه فرمانها کاربر را با معنای فیزیکی برای انجام دادن این عملیاتها آشنا می کند .

Obj کنترلی مدل کاربر معمولاً شامل دو نوع Obj می باشد آنهایی که ذاتاً وابسته به درخواست و آنهایی که هدفشان شرکت در کنترل برنامه می باشد فرمهای قدیمی تر Obj ها ، Obj ذاتی و دومی را Obj کنترلی می نامند با نوعهای مختلفی از Obj کنترلها در بخش تکنیکهای متقابل در فصل دوازده مواجه خواهیم شد.

آنها دارای انتخاب نیروهای خطی ، مقایسه و راهنما و بسیاری از چیزهای دیگر از قبیل محصول مصنوعی بر طبق انتخاب ما بر روی Obj ذاتی است سعی می کنیم که کنترل Obj ها را ، که رفتار آنها بصورت مستقیم سازگار و براحتی شبیه هم می باشند را انتخاب نمائیم .

اعمال بایستی در کنترل Obj بکار رود و مانند Obj ذاتی ، بنابراین اگر ما کاربر را بایک شبکه نمایش داده شده آماده کنیم می توانیم امیدوار باشیم که او دسترسی ورود و حذف را در فضای خودش بدهد Obj کنترل بنابراین به هر مجموعه از Obj و عمل ها اضافه می شود . ما باید مکلاحظه کنیم که تلاش ما برای نگهداری مدل کاربر ساده و یکنواخت باشد .

دستورات زبانی : دستورات زبانی چه متقابل چه غیر متقابل در بسیاری از سیستم های کامپیوتری قضاوت یافت می شوند شاید بیشترین استفاده دستورات زبانی کنترل GCL می باشد که در این جا یکدسته سیستم کامپیوتر آموزش داده شده که چگونه هر وظیفه را در یک دسته پردازش می کند . که دستور شبیه زیر می باشد .

Job DDRUN CORE = 10000 TIME = b00 "

بسیاری از سیستمهای مشترک متقابل با مجموعه دستورهای یکسان انجام می شود ، هر دستور کاربر ، شامل یک یا چند کلمه یا کاراکترها ، متخصصین می کند اعمالی را که بای انجام شود . کاربر دستورها را با فشار دادن یک کلمه پایانی مانند Return و ESC که اعمال طلب می کند کامل می کند .

رابطه گرافیکی به ندرت شامل این قبیل دستورات می باشد . در عوض کاربران به بخش منو اشاره می کنند یا دکمه ای را فشار می دهد و یا کاراکتری را می کشد برای اینکه برنامه تشخیص دهد .

بعضی اوقات دستورات بندرت آشکار هستند کاربر یک موقعیت پیوسته بعضی اوقات اعمال را طلب می کند و بعضی وقتها فقط اطلاعات را فراهم می کند . به شرایط غالباً پیچیده می شود هنگامی که از چند وسیله ورودی مختلف استفاده شود . صفحه کلیدهای حروفی و رقمی که بعضی وقتها کلید های تابعی ساختار دستور زبان های گرافیکی بکار می روند سخت تر از طراحی نوع گرافیکی می باشد . کلیدهایی که برای بعضی از حل مشکلات تا تشخیص بکار می رود یک دستور زبان بیش از یک لیست از دستورات می باشد .

یک زبان واقعی قابل حس تر و محجسوس تر می باشد . که دستورات انتخاب کننده به یکدیگر به روش غیر مستقیم وا بسته باشد وز بان دستورات Syntax را تعیین می کند . بدن توجه به Syntax های غیر مشابه اولین شکل در طراحی یک مجموعه دستور ، مشکل طراحی زبان برای خودش می باشد ما باید یک نحو را تعریف کنیم و مطمئن شویم که این Syntax ها سازگاری دارند . Error های تکنیکی که کاربر می تواند به وجود بیاومرد را مختص نمائیم و توانایی جوابگویی به احتیجات کاربر را داشته

باشیم . و مطمئن شویم که Syntax ها در برنامه بطور کافی جامع و کامل هستند البته مشکلات ذکر شده برای کسی که اطلاعاتی راجع به طراحی زبان برنامه نویسی را دارد کاملاً آشنا می باشد . شامل تجربه در حل کردن این مشکلات قبل از طراحی یک دستور زبان موفق مورد نیاز می باشد .

شرح دادن منحنی یک دستور زبان خیلی نزدیک به مدت کاربر است . فرمان هاردی Obj های مدل کاربر اجرا می شود . این ارتباط کار طراحی دستور زبان را برای یک مدل کار خوب صادر می کند . هر فرمان به یک Act مربوط میشود . عملوندهای هر یک از فرمانها هستند که روی Obj ها تأثیر می گذارد . عملوند را قبل از مدل کاربر که بطور کامل تعریف شود شروع می شود زبانی که ما طراحی می کنیم احتمالاً توسط مردمی بی تجربه در امور برنامه نویسی هم ممکن است استفاده شود .

و غالباً اینطور می باشد . برنامه های کامپیوتری تعاملی ممکن است براس بار اول برای آنها عاری از کیفیت باشد . و پیچیدگی قابل توجهی داشته باشد . و آنها مجبور اند یاد بگیرند برای اینکه قادر باشند از برنامه استفاده کنند و ما می توانیم با ساختن دستوراتی آسانتر این وضعیت را سبک کنیم .

چون اغلب فرمانها ممکن است برای یک کاربر تازه کار کامپیوتر بسیار گیج کننده باشند .

Obj ها و Fact ها ، مدل کاربر را فراهم می کند . و به شروغع منایسب را برای طراحی دستور زبان اشاره می کنند ، بطور کلی در فرمهای انتزاعی و خلاصه شده از عملکردها تعریف می شود . ما قادر هستیم دستور زبان را بعنوان نمایش و وامقعی از مدل کاربر ببینیم در این عملیات ها که توسط این دستورات و Obj ها جایگرزین شده اند بطوریکه تعریف درستی از مدل کاربر را می دارد . و ما باید قادر باشیم بطور مستقیم از زبانهای برنامه نویسی نتیجه بگیریم آنهم در تمرین کردن و ممارست . ما باید تصمیم بگیریم که چه چیزی میسر است به دربر داشتن یک قابلیت انتخاب چند گانه در زبان برنامه نویسی .

اغلب مدل کاربر تغییرات قابل توجه را همانند نتیجه تصمیمات و طراحی دستور زبان متحمل خواخد شد . چطور می توانیم راجع به طراحی زبان برنامه تصمیماتی را بگیریم قدم اول اینست که باید شماری از کلیدهایی که در کارطراحی پیچیده هستند را درک کنیم یکسری از این تصمیمات اینها هستند .

طریقه فرمان دادن : تعدادی از برنامه های دارای تأثیر به یکدیگر اعمال کاربر را تفصیر و ترجمه می کند آنهم در دو یا چندین طریق مختلف مطابق را حالت برنامه . متن ویراتاری برای مثال : ممکن است حرف D بعنوان عمل پاک کردن Deleting تفصیر شود . در طول یک حالت در قسمتی از تایپ کردن در متن در حالتهایی دیگر

هر کدام از این حالتها که عملکردها می دهند توسط یک کاربر ترجمه شده است که طریقه ، دستورات نامیده می شود .

روشهای تک دستوره زبان از این مشکلات کاملاً دوری می کند دنباله انتخاب و بیشتر کاربردها مستلزم دستور عملوندهای انتخابی هستند .

مثلا عملیات انتخاب و پاک کردن ممکن است بعنوان یک عملوند مستقل تعیین شود .

که اگر انتخابی موجود باشد این انتخاب مهم باشد.

۳. Error های دستی:طرحی دستور زبان باید به رسیدگی کند به اینکه چطور یک برنامه باید در برابر ورودی های نادرست و بی فعلی واکنش نشان دهد. کاربر باید از این خطاها آگاه شود. این خیلی مهم است که کاربر از خطاهایی که انجام داده یک برنامه قطع شود در غیر این صورت که سلسله مراتب این فرمان های نادرست ادامه دهد که منجر به برنامه ای پر از error می شود.و چرخه فرامین بوسیله این دستورات نادرست باطل می شود.

استیل های دستور زبان:

نگه داشتن سازگاری در دستور زبان بسیار مهم است ما نباید بطور پایدار از کاربر را از یک محوطه ورود به تعبیر مجموعه ها انتقال دهم تا سر در گم شود اگر ما نحوه تک

دستوری را برای یک دستور بکار ببریم شاهد بهتر باشد ما باید مطمئن شویم که همه فرمانها با هم به نحوی متفق می باشند. و به اینکه ممکن یک استثنا در آن ها ما را گیج کند. در کل دستور زبان ها و فرامین باید با هم سازگاری داشته باشد و این امر باید واضح باشد تا کاربربتواند نحو فضایی دستورات را پیش بینی نماید.

در واقع سازگاری به این صورت بوجود آمده است که در آغاز توسط یک style مخصوص از دستور زبان تعریف شده و قبول شده است. دراین مقطع ما باید شماری از style های گوناگون را مطرح کنیم. که هر کدام برای دستور زبان های سازگار، اساس خاصی را تعریف می کنند. Style مورد انتخاب ما به شماری از فاکتورها-پیکر بندی سخت افزاری- میزان و حدی از obj ها و fact ها در مدل کاربرد ... وابسته خواهد بود.