معرفی زبان استاندارد مدلسازی UML

منبع:
Version 1.3, June 1999 ,OMG Unified Modeling Language

۱.مقدمه:

UML(Unified Modeling Language) مستقیماً روشهای Booch ، Rumbaugh و Jacobson را با هم ادغام کرده است، اما محدوده آن بیشتر از سه متدلوژی فوق است. UML همانطور که از نامش پیدا است یك زبان مدلسازی است تا یك متدلوژی. بطور معمول، هر متدلوژی شامل حداقل یك زبان مدلسازی و یك پروسه ساخت است.زبان مدلسازی شامل نمودارهایی است که هر متدلوژی برای نمایش تحلیل و طراحی سیستمها از آن استفاده میکند. اما یك پروسه ساخت شامل دستورات، راهنمایی ها و قدم های لازم برای انجام تحلیل و طراحی سیستمها میباشد. افرادی که یك متدلوژی را به کار می برند معمولاً بیشتر با زبان مدلسازی آن سروکار دارند.هدف طراحان UML بیشتر تدوین یك زبان مدلسازی شیگرا بوده تا ارائه یك پروسه ساخت استاندارد، اگر چه طراحان UML یك پروسه ساخت به نام USDP(Unified Software Development Process) را نیز ارائه دادهاند.

۲- تاریخچه:

UML یك زبان استاندارد برای نمایش، ایجاد و مستندسازی سیستم های نرمافزاری مبتنی بر روشهای شیگرا میباشد. قبل از UML نیز روشهای شیگرایی متعددی توسطافرادمختلف برای مدل سازی سیستمهای شئگرا ارائه شده بود. اتفاقی که باعث ایجاد UML شد بدینصورت بود که Rumbough ، طراح متدلوژی OMT به شرکت Rational که متعلق به Booch بود پیوست و آنها تلاش خودرا برای ایجاد یك زبان مدلسازی شیگرای متحدالشکل بکار گرفتند. ترکیب دومتدلوژی و ایجاد زبانUML اعتبار ویژهای به آن بخشید. درسال ۱۹۹۵،شرکت Rational آماده بود تا اولین مستندات UML(نسخه ۰,۸) را ارائه نماید، اما در یك اقدام ناگهانی امتیاز شرکت Jacobsonرا که مالك Objectory بود، خریداری نمود. پس از این اقدام، شرکت Rational با ترکیب سه متدلوژی سطح بالا قادر به ارائه یك استاندارد در روشهای شیگرا بود.

در سال ۱۹۹۷ ، UML بعنوان یك زبان استاندارد مدلسازی شیگرا از طرف گروه OMG(Object Management Group)پذیرفته شد. مهمترین قابلیت این زبان ارائه مدلهایی بصورت دیاگرام برای کل چرخه حیات نرم افزار است و میتواندبصورت یك زبان ارتباطی بین تمام گروههای یك تیم پروژه استفاده شود. از قابلیتهای دیگر آن اینست که سازگاری خود را با اغلب روشهای متداول مانند OMT، Booch وOOSE حفظ نموده است.
از دید مستندسازی، UML قادر است کل چرخه حیات سیستم را در قالب نمودارهایی بصورت کلی و قابل فهم ارائه نماید که میتواند مستقل از متدلوژی ساخت ارائه شود هرچند که برخی از متدلوژیها دیاگرامهای خاص خود را دارند. اما با توجه به نزدیکی متدلوژیهای شئگرا و شباهت دیاگرامهای آنها میتوان UML را در بسیاری از متدلوژیهای شئگرا استفاده نمود. شرکتهایی مانندUnisys, Rational, Oracle, IBM, Microsoft, HP و ... از شرکتهایی هستند که ازUML استفاده کرده و آن را پشتیبانی مینمایند.

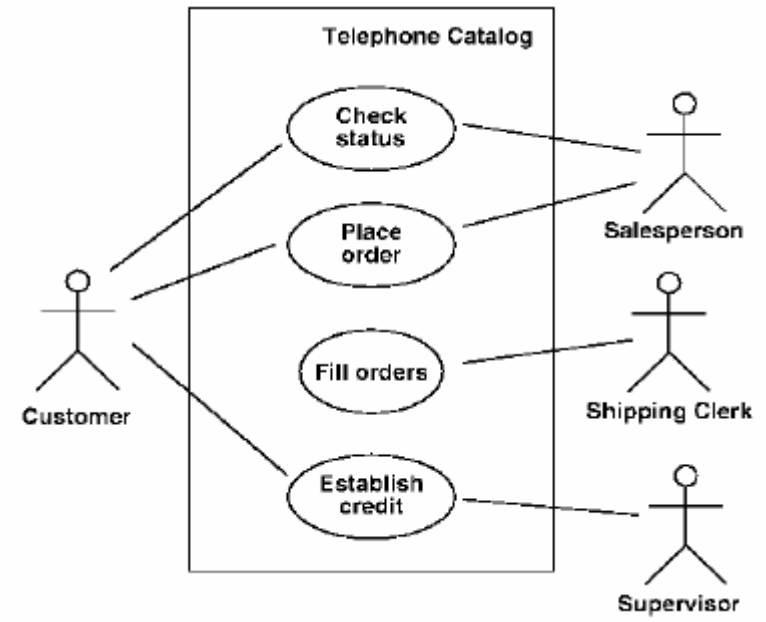
۳- دیاگرام های UML یکی از بزرگترین اهداف طراحی برنامه های سیستم نرم افزاری ایجاد برنامه های صحیح است به نحوی که نیازهای کاربران را بدرستی و با هزینه قابل قبولی برآورده نماید. فهم نیازهای کاربران که مستلزم ایجاد ارتباط با آنهاست، یکی از نکات کلیدی در ایجاد نرم افزارهای مفید میباشد. روشی که در UML برای نشان دادن این خواستهها بکارگرفته میشودUse Case نام دارد. مجموعه تمامیUse Case ها، تصویر خارجی سیستم را تشکیل میدهد. یك مجموعه خوب از Use Case ها وقتی حاصل خواهد شد که طراح بداند کاربران چه چیزی از سیستم میخواهند. Use Case ها همچنین ابزار خوبی برای پیشبرد پروژه میباشند زیرا ساخت سیستم به طریق تکراری را کنترل میکند. در واقع از قابلیتهای ویژه متدلوژیهای شئگرا این است که میتوان یك مجموعهای از نیازها را انتخاب نموده و طراحی و پیادهسازی آنرا به پیش برد و بدین صورت ساخت سیستم بهصورت افزایشی را مقدور میسازد همچنین با توجه به این که تیم طراح و برنامهنویس در هر مرحله با Case Use ها سروکار دارد، در هر مرحله امکان بازگشت سریع به مراحل قبل وجود دارد.

این روش خود تکنیك با ارزشی است، زیرا نتایج را مرحله به مرحله به کاربر باز خور میدهد. در ادامه این بخش، دیاگرامهایی که در UML برای مدل سازی سیستم های نرم افزاری استفاده میشود به صورت اجمالی معرفی میگردد.

۳-۱- دیاگرام Use Case

در ساده‌ترین حالتUse Case ها بوسیله مصاحبه با کاربران و انتظاراتی که آنها از سیستم دارند تعریف می‌شود. برای نمایش اطلاعات مربوط به نیازهای کاربران که در قالب Use Case هاجمع‌آوری شده است از دیاگرام Use Case استفاده می‌شود. شکل شماره ۱ نمونه‌ای از این دیاگرام را نشان می‌دهد

شکل شماره ۱ - نمونه‌ای از دیاگرام Use Case



نمادهای آدمك که در شکل دیده می‌شود، بازیگر (Actor)نامیده می‌شوند. هر بازیگر می‌تواند یك کاربرسیستم باشد که نقش یا نقش‌هایی را در سیستم بازی می کند. یك بازیگر لزومی ندارد حتماً یك انسان باشد، یك سیستم خارجی نیز میتواند یك بازیگر باشد.

چهار نوع رابطه در دیاگرام Use Case وجود دارد که عبارتند از :

الف- رابطه Communication : نشان می‌دهد یك بازیگر، یك Use Case را استفاده می‌کند.

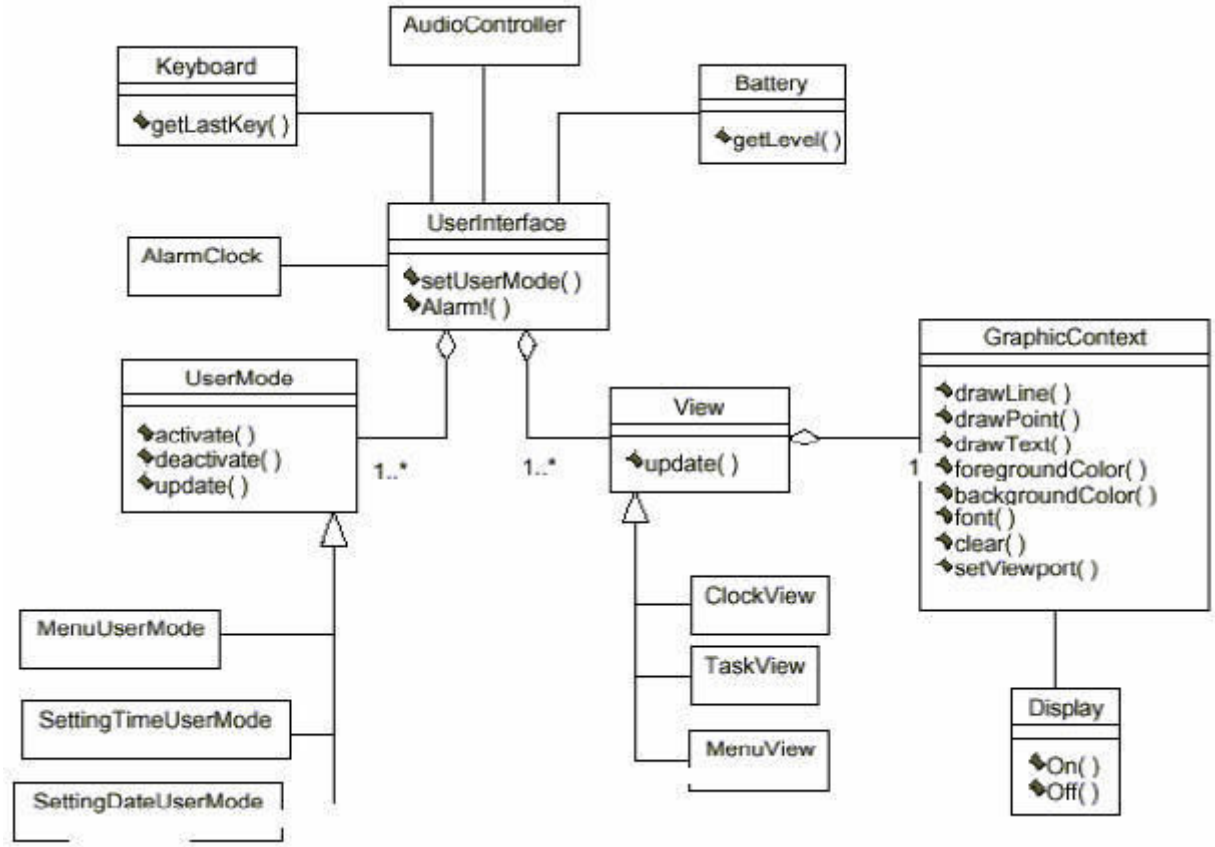
ب- رابطه Extend : ارتباط بین دو Use Case که حالات خاص یکی در دیگری قرار داده می‌شود.

ج- رابطه Uses : شبیه Extend می باشد ولی در اینجا حالات خاص‌را در یك Use Case قرار نمی‌دهد، بلکه اگر رفتاری در چند Use Case مشترك باشد، آنرا جدا کرده و در یك Use Case مجزا قرار می‌دهند.

۳-۲ - دیاگرام کلاس : دیاگرام کلاس یکی از دیاگرامهای مهم و اساسی در متدلوژیهای شیگرا می‌باشد که هر متدلوژی حالات مختلفی از آنرا استفاده می کند. دیاگرام کلاس شامل اشیاء و روابط مابین آنها می‌باشد. همچنین دیاگرام کلاس شامل صفات و رفتار کلاس‌ها می‌باشد.

در دیاگرام کلاس اگر فقط اشیاء (یعنی نمونه‌های کلاس ها) و روابط آنها نشان داده شود، آن را دیاگرام شیء (Object Diagram)گویند. شکل شماره ۲ نمونه‌ای از دیاگرام کلاس را نمایش می‌دهد. در ادامه مفاهیمی که در دیاگرام کلاس استفاده می‌شود بطور مختصر شرح داده می‌شود.

شکل شماره ۲- نمونه‌ای از دیاگرام کلاس



۳-۲-۱- کلاس:

کلاس، مشخصه‌ای برای مجموعه‌ای از اشیاء با صفات، رفتار و روابط مشابه می‌باشد. UML برای نمایش کلاس از نماد مستطیل سه قسمتی استفاده می‌کند. قسمت فوقانی این نماد برای نمایش مشخصات نام کلاس، قسمت میانی برای نمایش مشخصات صفات و قسمت تحتانی برای نمایش مشخصات رفتار و اعمال کلاس بکار می رود. برخی عناصر که از نظر مفهوم شبیه کلاس هستند نیز از همین نماد و خواص استفاده می‌کنند.

کلاس‌های مشتق شده از یك کلاس معمولاً با مفهوم کلیشه (Stereotype)بیان می‌شود. این مفهوم در UML استفاده زیادی دارد و حالات خاص و مختلف از یك چیز را نشان می‌دهد. انواع مختلفی که از نظر مفهومی شبیه کلاس هستند با ذکر یك کلیشه، در قسمت نام کلاس مشخص می‌شوند. سه نمونه از این مفاهیم، تایپ (Type)، کلاس پیاده‌سازی(Implementation Class)، و واسط (Interface Class) نام دارند.

تایپ، نشان‌دهنده یك نقش قابل تغییر است که یك شئ می‌تواند انتخاب کند و سپس آنرا کنار بگذارد.

یك کلاس پیاده‌سازی، نشان‌دهنده ساختار فیزیکی و توابع یك شئ است که در یك زبان، پیاده سازی می‌شود و قابل تغییر نیست. یك شئ می تواند دارای چندین تایپ باشد اما فقط یك کلاس پیاده سازی دارد. بین یك تایپ و یك کلاس پیاده سازی می توان رابطه Realize برقرار نمود که نشان می‌دهد یك تایپ توسط کدام کلاس پیاده سازی شده است.

یك واسط نشان دهنده اعمال یك کلاس است که قابل رویت توسط دیگران است. واسطها فاقد صفت،حالت و روابط انجمنی هستند و فقط شامل اعمال می‌باشند. واسطها می توانند روابط کلی/ اختصاصی داشته باشند.

کلاس پارامتری(Parameterized Class) معرف کلاسی است که تعدادی پارامتر داشته ولی می‌تواند خانواده‌ای از کلاسها را تعریف کند. هر کلاس از این خانواده با تعریف مقادیر واقعی پارامترها، مشخص می‌شود. یك کلاس پارامتری با همان نماد کلاس نمایش داده می‌شود. کلاسهایی که به یك کلاس پارامتری محدود یا مقید می‌شوند، عناصر مقید (Bound Element) نامیده می‌شوند.

شئ نمونه‌ای از یك کلاس است که صفاتش دارای مقادیر حقیقی هستند. نماد شئ مانند نماد کلاس است که در زیر نام آن خط کشیده می‌شود. یك شئ مرکب (Composite Object)شئ سطح بالایی است که در درون خود دارای شئ هایی است که این شئ ها می توانند دارای روابط فی مابین باشند.

۳-۲-۲ روابط مابین کلاسها و اشیاء : در زیر به انواع روابط بین کلاسها و اشیاء اشاره می‌شود.

الف- رابطه انجمنی :

یك رابطه انجمنی می‌تواند بین دو کلاس و یا چند کلاس برقرار شود. مشخصات کلی این رابطه توسط چند خصیصه تعیین می شود که عبارت است از مقدار کثرت، ترتیب، علامت جهت و مشخصه اجماع (Aggregation)

ب- رابطه در برگیری (Composition [Whole-Part])

این رابطه یك رابطه قویتر از رابطه اجماع می‌باشد که در آن وجود کلاس جزء وابسته به وجود کلاس کل می باشد. رابطه دربرگیری با یك خط راست که در طرف کلاس کل یك لوزی توپر قرار دارد نمایش داده می‌شود.

ج- رابطه کلی- اختصاصی(Generalization / Specialization)

رابطه‌ای است که بین یك کلاس عمومی و کلاسهای اختصاصی آن کلاس برقرار می‌شود. در این نوع رابطه، کلاس های اختصاصی تمامی خصوصیات کلاس عمومی را به ارث برده و علاوه برآن می‌توانند یکسری خصوصیات دیگر را نیز تعریف نمایند. این رابطه را رابطه ارثبری نیز می‌نامند.

د- رابطه وابستگی این رابطه یك رابطه معنایی را بین دو عنصر تعریف می‌کند که تغییر در یك عنصر ممکن است باعث تغییراتی در عنصر دیگر شود. ازا نواع این روابط میتوان به bind , uses اشاره نمود.

هـ- رابطه Realize

رابطه‌ای است که بین کلاس پیاده‌سازی و تایپ برقرار می‌شود.

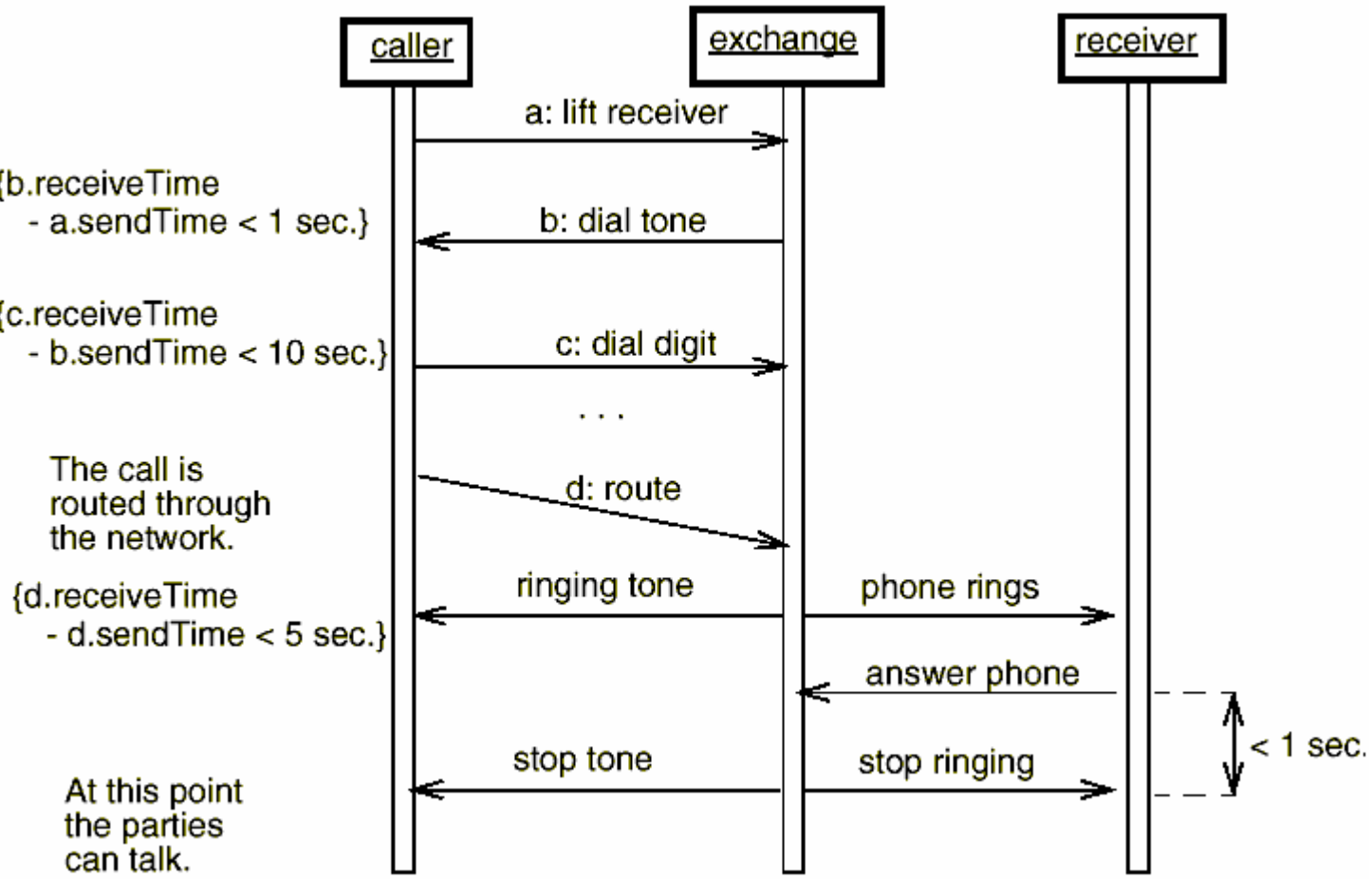
۳-۳- دیاگرام تعامل :

دیاگرام تعامل (Interaction Diagram) بیانگر رفتار اشیاء در روابط با یکدیگر است. معمولا هر دیاگرام تعامل، رفتار یك Use Case را نشان می‌دهد. دیاگرام تعامل خود به دو نوع دیاگرام ترتیب (Sequence Diagram) و دیاگرام همکاری (Collaboration Diagram) تقسیم می‌شود.

الف - دیاگرام ترتیب :

یك نمونه از این دیاگرام در شکل شماره ۳ نشان داده شده است. عملیات بصورت ارسال پیام ما بین اشیا انجام می‌گیرد. ترتیب ارسال پیام از بالابه پایین می‌باشد. هر پیام دارای یك نام بوده و می‌تواند پارامتر و اطلاعات کنترلی نیز داشته باشد. دو نوع اطلاعات کنترلی میتواند درکنار پیام نوشته شود، یکی شرط ارسال است که با بر آورده شدن آن پیام ارسال می‌شود و دیگری علامت تکرار است که مشخص می کند پیام چند مرتبه به اشیا فرستاده می‌شود.

شکل شماره ۳ - نمونه‌ای از دیاگرام ترتیب

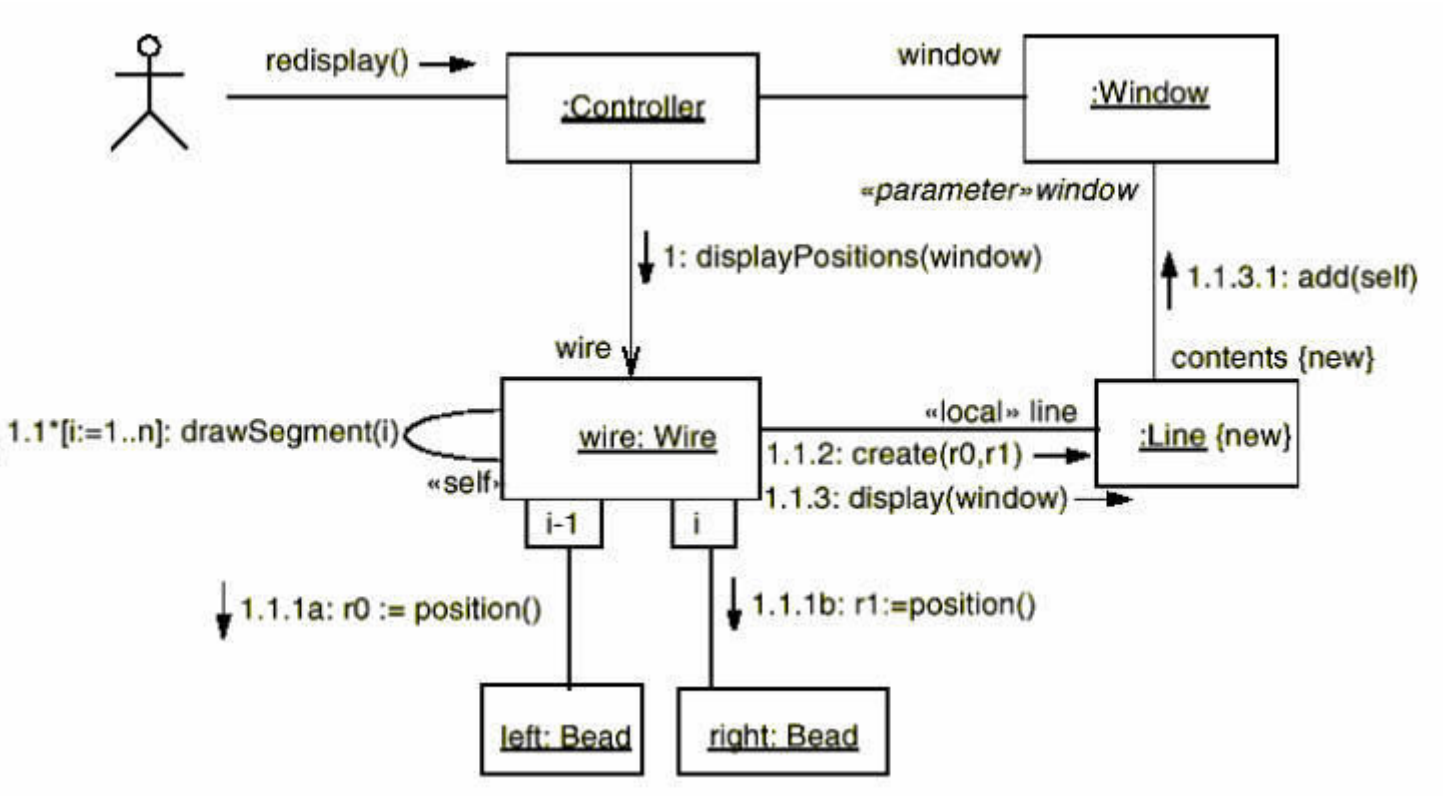


ب - دیاگرام همکاری

در دیاگرام همکاری، ارتباط بین اشیا بدون درنظر گرفتن ترتیب، نشان داده می‌شود. نمونه‌ای از این دیاگرام در شکل شماره ٤ نشان داده شده است.

شکل شماره ٤- نمونه‌ای از دیاگرام همکاری

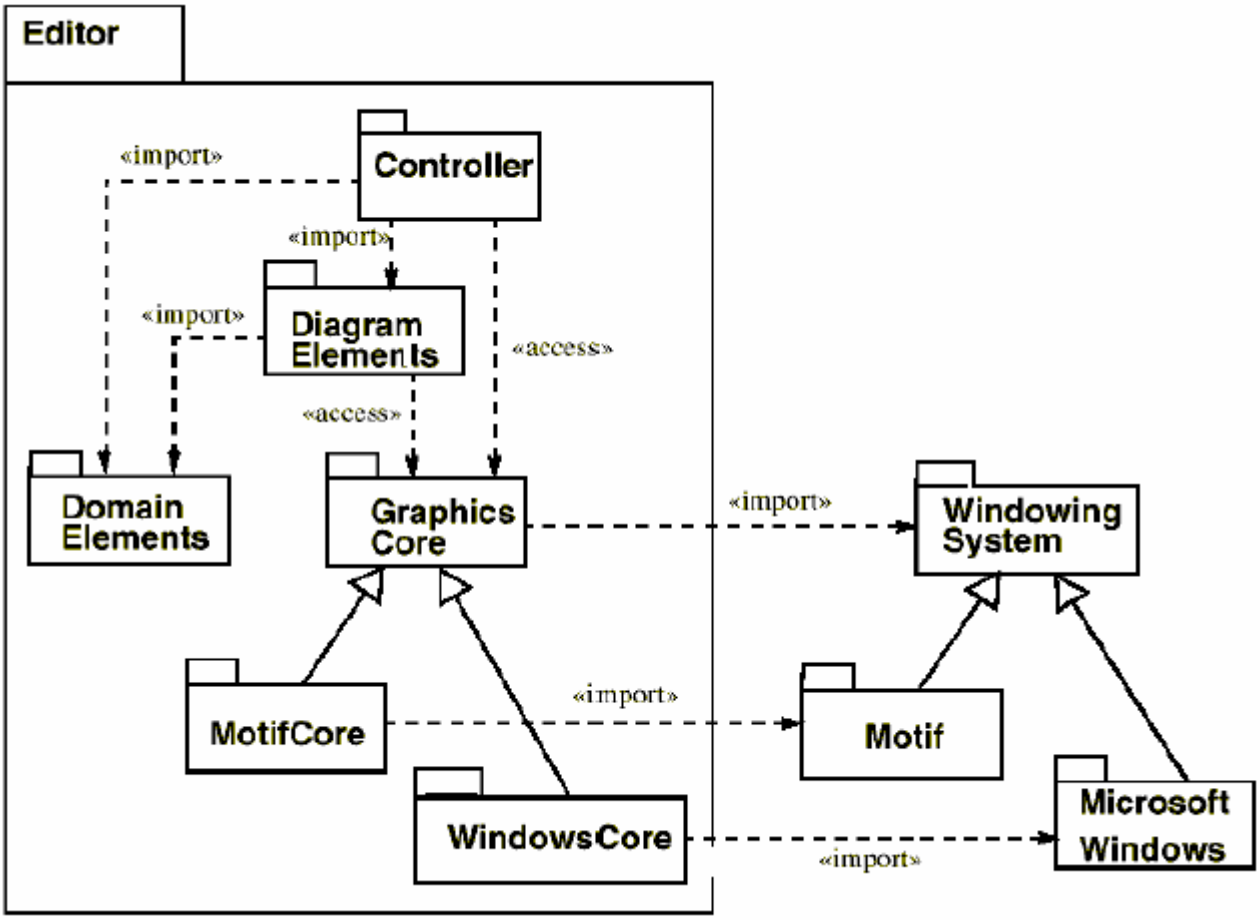
!Error



٣-٤- دیاگرام بسته (Package Diagram)

دیاگرام بسته برای تجزیه یك سیستم بزرگ به اجزاء کوچکتر استفاده می‌شود. این دیاگرام عین دیاگرام کلاس می‌باشد. در این دیاگرام چندین کلاس در قالب یك واحد سطح بالاتر به نام بسته قرار داده می‌شوند. می‌توان بصورت ترکیبی دریك دیاگرام، بسته‌ها و کلاس‌ها را با هم نمایش داد. بین بسته‌ها میتوان روابط وابستگی برقرار کرد. نمونه‌ای از یك دیاگرام بسته در شکل شماره ٥ ، نمایش داده شده است.

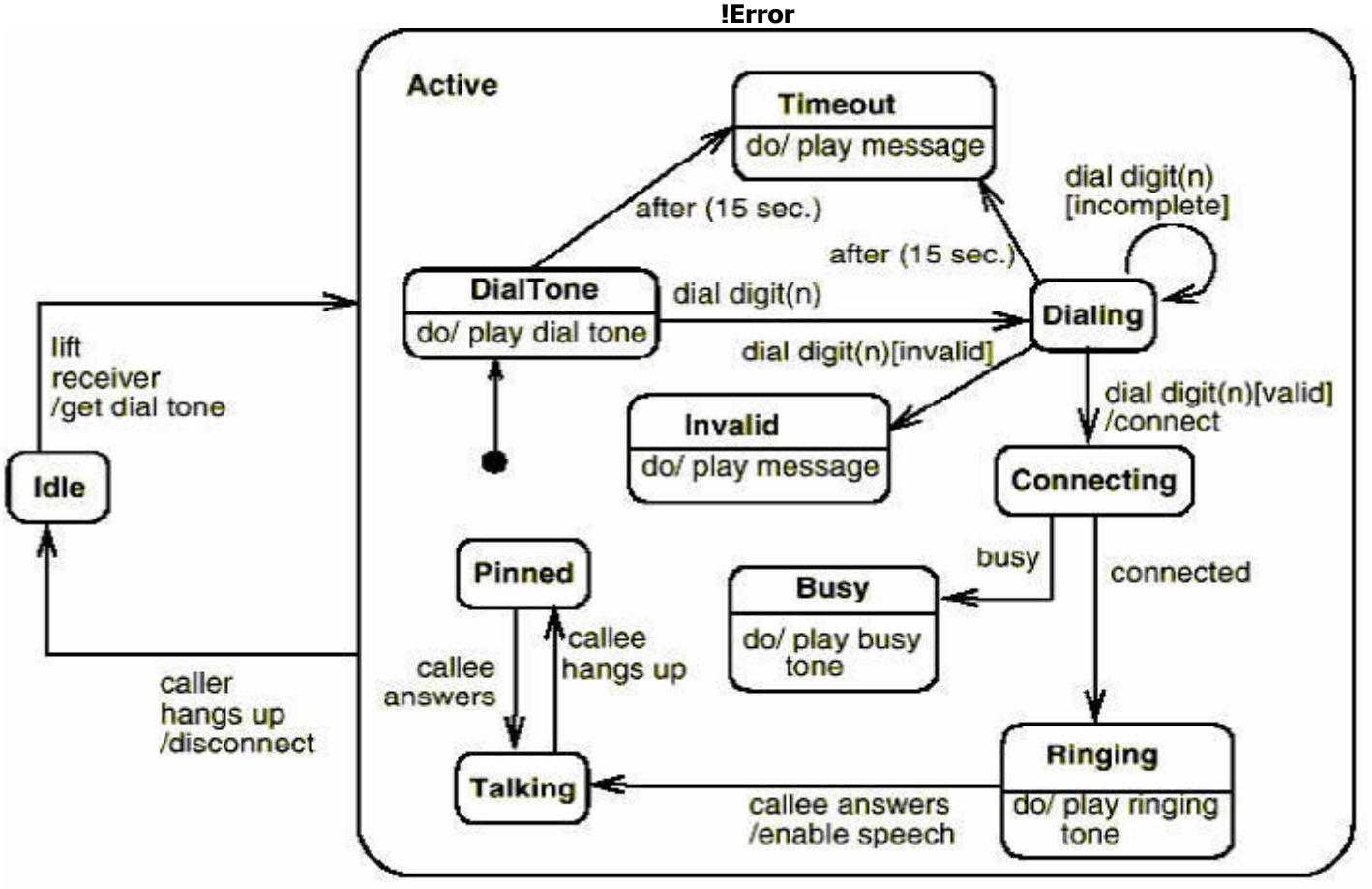
شکل شماره ۵ - نمونه‌ای از دیاگرام بسته



۳-۵- دیاگرام حالت (State Diagram)

دیاگرام حالت یکی از دیاگرامهای مرسوم در توصیف رفتار سیستم می باشد. از دیاگرام حالت می‌توان برای نمایش حالات مختلفی از نمونه های کلاس استفاده نمود. هر حالت با نماد مستطیل خوابیده با گوشه های مدور نمایش داده می‌شود. در داخل مستطیل نام حالت نوشته می‌شود. همچنین اگر در یك حالت، عملی انجام شود در اینصورت به شکل Activity / Do نوشته میشود. انتقال بین حالتها می‌تواند همراه با عملی باشد که با فرمت "Event + [Guard] / Action" نوشته میشود. در آنجا Event همان رویدادی است که باعث انتقال می‌شود. Guard نیز شرطی است که در صورت برقراری آن، انتقال صورت میگیرد و Action عملی است که در حین انتقال صورت میگیرد. شکل شماره ٦ یك دیاگرام حالت را نمایش می‌دهد.

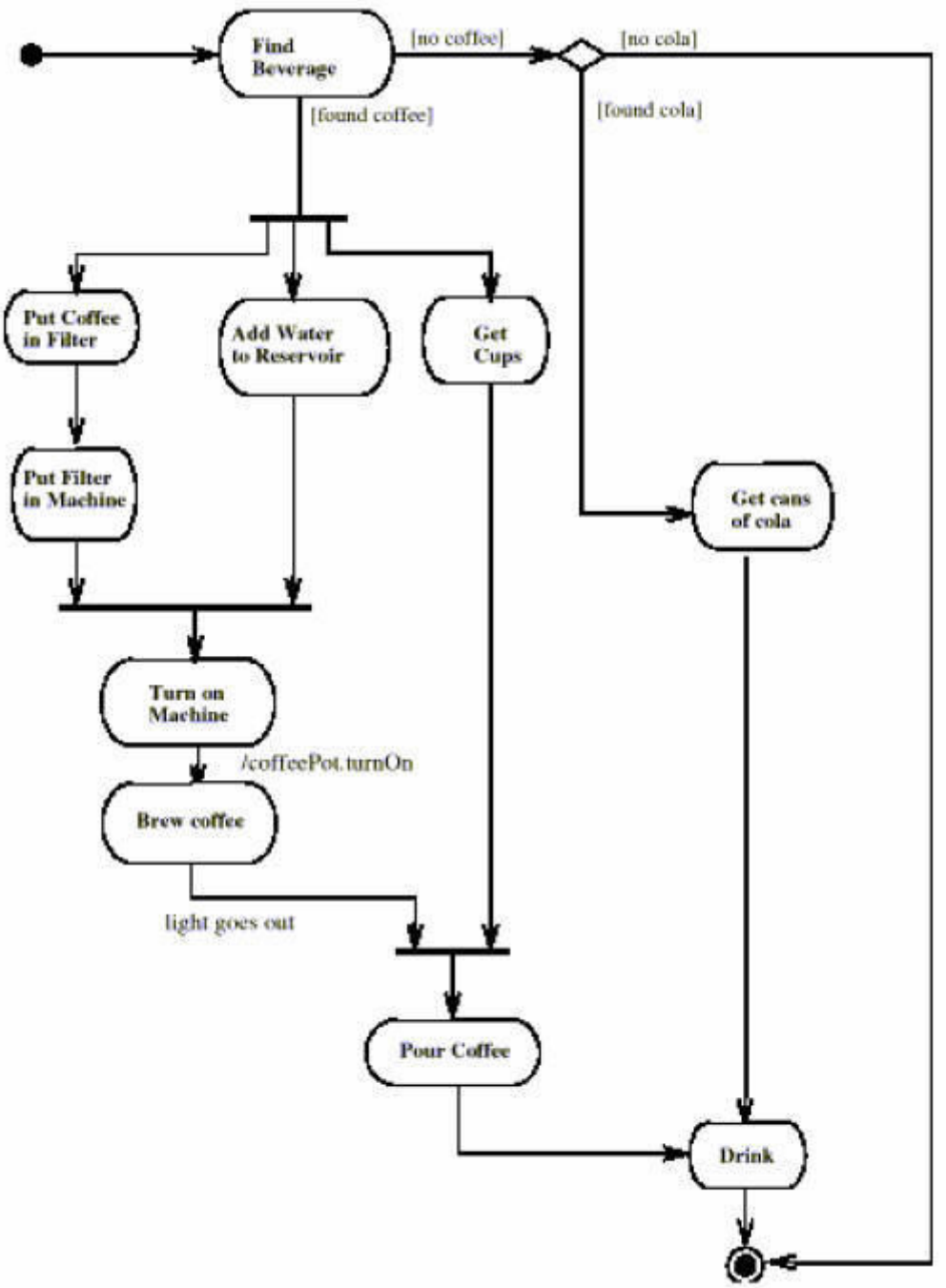
شکل شماره ٦- نمونه‌ای از دیاگرام حالت



٣-٦- دیاگرام فعالیت

این دیاگرام همانند فلوچارت می باشد با این تفاوت که در فلو چارت تمامی اعمال بصورت ترتیبی ذکر می‌شود. ولی در اینجا می توان فعالیتهای موازی را نمایش داد. دیاگرامهای فعالیت برای مدل کردن رفتارهای(متدهای) پیچیده استفاده می‌شود. شکل شماره ٧ نمونه‌ای از این نوع دیاگرام را نمایش می‌دهد.

شکل شماره ۷- نمونه‌ای از دیاگرام فعالیت

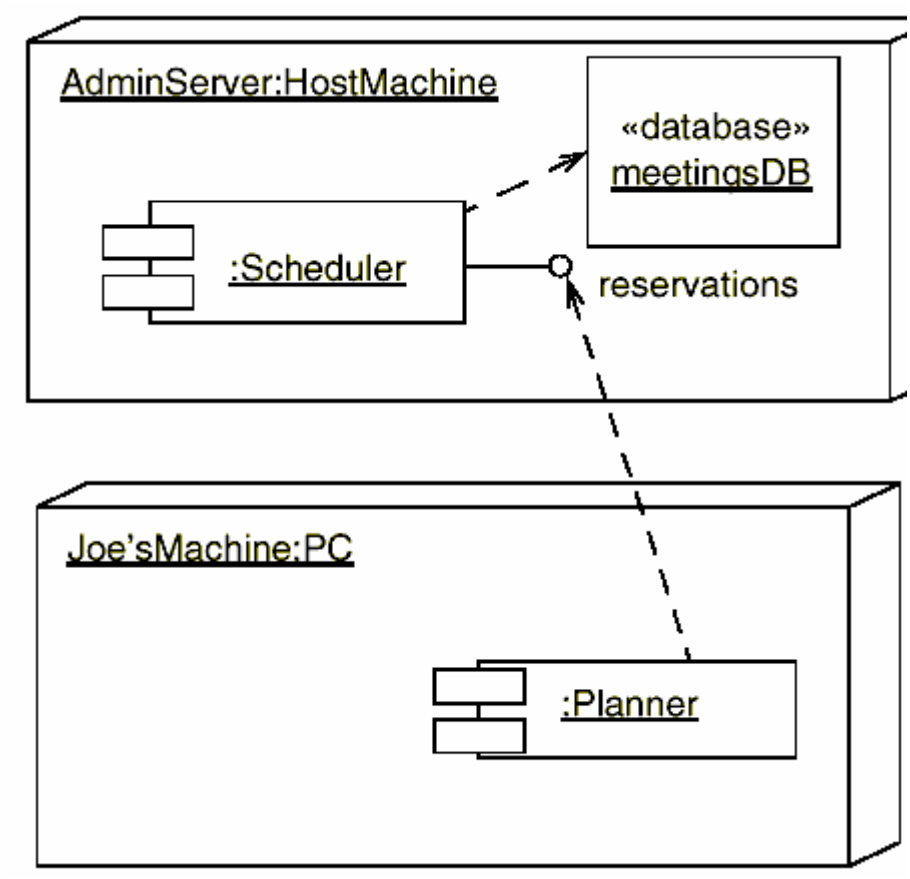


در این دیاگرام انتقالها از بالا به پایین صورت می‌گیرد. علامت * در نمودار بدین معنی است که فعالیت مورد نظر برای چندین عنصر میتواند بصورت موازی انجام شود. خطوط افقی برای سنکرون کردن عملیات موازی استفاده می‌شود

۱. ۳-۷- دیاگرام آرایش قوا (Deployment Diagram)

این دیاگرام رابطه بین اجزاء نرم‌افزاری و سخت‌افزاری را نشان می‌دهد. این دیاگرام محل خوبی برای نمایش انتقال اشیاء و اجزاء در یك سیستم توزیع شده می‌باشد. هر گره (Node)در دیاگرام نشان دهنده یك واحد محاسباتی می‌باشد. شکل شماره ۸ نمونه‌ای از این دیاگرام را برای ارتباطات بین شبکه‌ای یك PC و یك سرویسگر Unix نمایش می‌دهد.

شکل شماره ۸ - نمونه‌ای از دیاگرام آرایش قوا



صمیمانه منتظر انتقادات و پیشنهادات شما هستم.

آرش نوری
Arashmidos2006@gmail.com
Http://cplusplus.blogfa.com