راهنمای توسعه و نگهداری نرم افزارکال منجر[1]

شرکت کامپیوتر و ارتباطات پیشرفته بسامد

بهار ۱۳۸۶

بهراد زاری

behradz@gmail.com

1 OPXI Call Manager

فهرست

۱ معرفی

نرم افزار *کال* منجر یکی از زیر سیستم های اساسی سیستم یکپارچه مرکز تماس OPXI به حساب می آید و نقش اصلی آن برقراری، مدیریت و کنترل مکالمات و اپراتورهای مراکز ارتباطی می باشد. نرم افزار کال منجر در واقع دو گروه سرویس عمده به دنیای بیرون خود ارایه می دهد. یک گروه سرویس های اپراتوری می باشند که شامل سرویس هایی از قبیل رجیستری، اعلان وضعیت و پیغام متنی هستند و گروه دیگر شامل سرویس های مکالمه از قبیل پراکسی مکالمات، سرویس صف برای تماس با اپراتور، انتقال مکالمه، کنفرانس و... هستند.
نرم افزار کال منجر که بر اساس اخیرترین تکنولوژی های ارتباطی[2] طراحی و پیاده سازی شده خود شامل مؤلفه های درونی مهمی است که با تکنولوژی های مختلف پیاده سازی شده اند. در انتخاب تکنولوژی ها، معیار استاندارد بودن بیشترین اهمیت را داشته و در طراحی مؤلفها قابلیت گسترش ملاک بوده است. راهنمای حاضر، برای افرادی با نقش برنامه نویس و یا توسعه دهنده سیستم نوشته شده است. در این راهنما (برای سادگی بیشتر) از آوردن جزییات پیاده سازی مربوط به داخل مؤلفه ها خودداری شده مگر در مواردی که اهمیت آن بالا بوده است.

۱.۱ پیشنیاز ها و تکنولوژی های مورد استفاده

برای طراحی و پیاده سازی نرم افزار کال منجر از تکنولوژی های مختلفی استفاده شده است که توسعه دهنده این نرم افزار می بایست دست کم با آنها آشنایی اولیه ای داشته باشد. بدین منظور تکنولوژی ها و استانداردهای مهم بکار گرفته شده را به همراه منابعی برای مطالعه بیشتر در مورد هر یک از آنها لیست کرده ایم.

آشنایی با Core Java: نسخه فعلی نرم افزار بر اساس J2SE 1.4 نوشته شده است هر چند که امکان استفاده از Java 5 نیز به سادگی وجود دارد. (امکان کامپایل و اجرای نرم افزار روی JDK 1.5.0_6 با موفقیت تست شده است.)

آشنایی با پروتکل SIP: یکی از استاندارد های اخیر برای برقراری ارتباط (متنی، صوتی و تصویری) و مکالمات روی شبکه های IP می باشد. شناخت مفاهیم پایه این پروتکل برای توسعه مؤلفه ی مربوط به آن ضروری است. برای مطالعه بیشتر می توانید به [۱] رجوع کنید.

آشنایی با محیط Servlet Programming و Struts: از آنجایی که این نرم افزار یک Converged SIP/Web Application است، امکان پیاده سازی یک محیط مبتنی بر وب را برای مدیریت و کنترل آن فراهم می- کند. این محیط فعلاً بر اساس چارچوب Struts پیش بینی شده است.

---

2 VOIP

آشنایی با چارچوب Sip Servlet API: مطرح ترین بستر برنامه نویسی بر اساس پروتکل SIP در محیط جاواست که در قالب چارچوب استاندارد و معروف Servlet API معرفی شده است. برای مطالعه بیشتر می توان به [۲] رجوع کرد.

آشنایی با نرم افزار های Ant و XDoclet: مراحل کامپایل و ساخت فایل قابل اجرای پروژه، به طور اتوماتیک توسط Ant انجام می شود. همچنین در مراحلی که امکان داشته کد جاوا و یا فایل های دیگری اتوماتیک تولید شوند از Xdoclet استفاده شده است.

آشنایی با چارچوب Hibernate: برای نگه داری موجودیت های ماندگار در پایگاه داده کال منجر از چارچوب Hibernate استفاده شده است.

آشنایی با زبان مدل سازی UML: برای نمایش نمودار های مدل کال منجر در بخش های مختلف، از نمودار های UML 2.0 استفاده شده است. بنابراین آشنایی اولیه با این زبان مدل سازی برای درک مدل کلی نرم افزار لازم است.

۱.۲ ساختار مستندات

در مستندات حاضر ابتدا با نگاهی سطح بالا جنبه های مختلف معماری کال منجر را تشریح خواهیم کرد و سپس هر جا که لازم باشد وارد جزییات خواهیم شد. اولین جنبه معرفی شده از معماری، ساختار(معماری) فیزیکی پروژه است که شامل ساختار فایلینگ آن می باشد. بعد از مرور ساختار فایلینگ، سراغ معماری منطقی پروژه می رویم و به نوعی طراحی داخلی کال منجر را از دید انواع سرویس ها و مؤلفه های موجود می بینیم و در انتها طراحی داخل هر مؤلفه را خواهیم دید. هر کجا لازم بوده، بسته به جنبه ای از طراحی که مورد اهمیت بوده از نمودارهای مربوطه استفاده شده است.

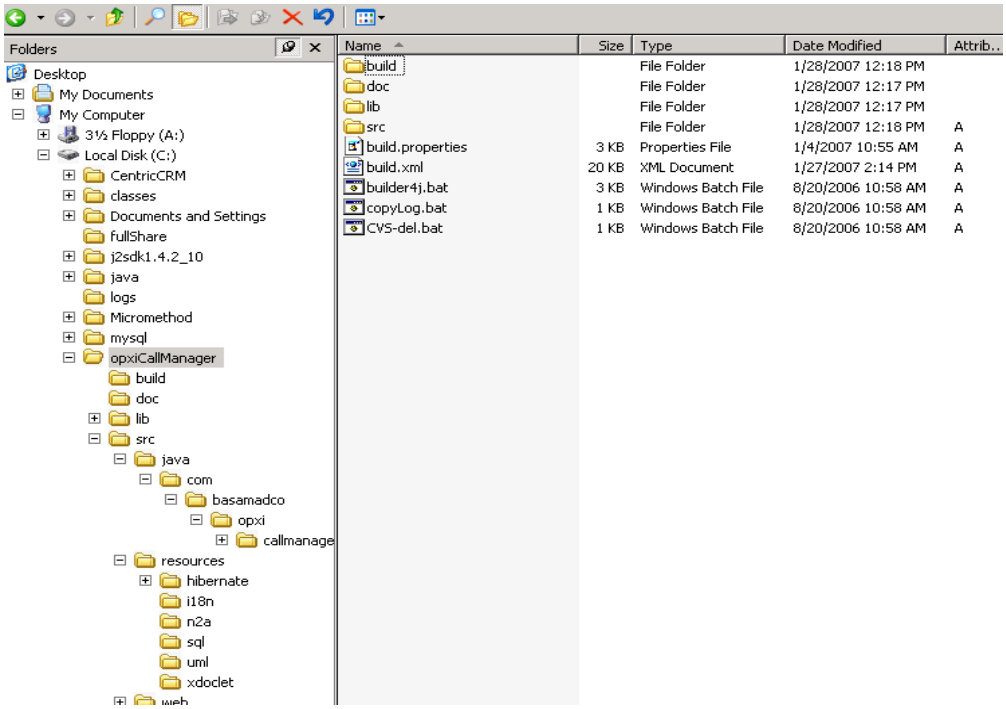
علاوه بر مستنداتی که در پیش روی شما ست، نرم افزار کال منجر حاوی مستندات و اطلاعات مفیدی در سطوح پایین تر می باشد که در قالب استاندارد Javadoc داخل کد آورده شده اند.

۲ معماری فیزیکی پروژه

معماری فیزیکی پروژه را از دو دیدگاه بررسی می کنیم. ابتدا ساختار شاخه ها و فایلینگ پروژه را خواهیم دید و بعد جایگاه و نحوه قرار گرفتن فایل اجرایی را در محیط سرور و در زمان اجرا خواهیم دید.

۲.۱ ساختار فایلینگ پروژه در زمان توسعه

مسیر ریشه پروژه کال منجر شامل شاخه هایست که حاوی فایل ها و منابع لازم در زمان توسعه، کامپایل و ساخت هستند. این مجموعه ساختار فایلینگ کال منجر را تشکیل می دهد. نمای کلی ساختار پروژه روی دیسک (در مسیر ریشه) مطابق شکل زیر است:



شکل ۱. نمای کلی ساختار فیزیکی پروژه کال منجر

توضیحی در مورد شاخه های موجود در پروژه در جدول زیر آمده است:

نام شاخه	توضیح
build	خروجی های عملیاتbuild و کامپایل در اینجا قرار می گیرند. تمامی محتویات این شاخه به صورت اتوماتیک قابل ایجاد هستند.
doc	شامل مستندات نرم افزار مثل راهنمای نصب، راهنمای برنامه نویس و... می باشد.
docs	مستندات Javadoc در این مسیر بصورت اتوماتیک ساخته می شوند.
lib	تمامی library ها در قالب فایل های جار مربوط به وابستگی های پروژه به پروژه ها و کتابخانه های دیگر

src	حاوی فایل های پیکربندی نرم افزار روی سرور مثل **sip.xml**، **application.xml** و **opxiCallManager.properties**
src/java	حاوی فایل های سورس کد برنامه (به زبان جاوا) می باشد که با توجه به نگاشت یک به یک package ها به شاخه ها در سیستم عامل، مطالعه بخش معماری منطقی، دید شما را نسبت به ساختار و ماهیت package های موجود روشن می کند.
src/resources	حاوی انواع فایل هایی که به نوعی برای build لازم اند. این منابع شامل فایل های XSD، اسکریپت های پایگاه داده ، فایل های حاوی پیام ها ی کاربری کال منجر در زبان های مختلف ، تنظیمات مربوط به XDoclet و ... می باشند.
src/web	حاوی فایل های JSP می باشد.

جدول ۱. لیست شاخه های موجود در پروژه

حال به شرح جزییات اجزای مختلف ساختار فایلینگ پروژه می پردازیم. در این قسمت سعی بر این بوده تا یک طبقه بندی منطقی از مسیر ها و فایل های موجود در پروژه ارایه کنیم به ترتیبی که فهم اجزای این ساختار ساده تر شود.

۱. خروجی عملیات کامپایل و ساخت

این ها مسیر هایی هستند که در حین انجام مراحل مختلف فرایند کامپایل و ساخت بطور اتوماتیک توسط این فرایند ساخته می شوند و همواره قابل ایجاد مجدد هستند. مسیر های build و docs جزو این گروه هستند.

۲. کتابخانه های مورد نیاز

این مسیر که lib می باشد، شامل کتابخانه های مورد نیاز است که در محیط جاوا همان فایل های جار[3] هستند. از آن جایی که تعداد فابل های جار مربوطه زیاد بوده، برای طبقه بندی و ساختاردهی منظم و ساده تر آن ها را بر اساس ماهیت وابستگی در مسیر های مختلف قرار دادیم که با مراجه به شاخه lib و مشاهده آن ها به راحتی به اینکه کدام جار فایل برای چه منظوری در پروژه قرار دارد، پی خواهید برد اما در این میان به چند مسیر مهم اشاره می کنیم.

3 Jar Files

۱.۲ lib/ant و lib/xdoclet: در این مسیر ها فایل های جار مربوط به اجرای مراحل کامپایل و ساخت توسط نرم افزار های Ant و XDoclet قرار دارند.

۲.۲ lib/ext: در این مسیر فایل های جاری قرار دارند که باید در محیط اجرا نیز داخل پکیج نرم افزار باشند. این فایل ها به همین ترتیب از این شاخه به پکیج نهایی کپی می شوند.

۳.۲ lib/XXX: این مسیر ها مجموعه بقیه مسیر هایی هستند که فایل های جار دیگر را در خود دارند. مثلا شاخه lib/hibernate فایل های جار مربوط به چارچوب Hibernate و یا مسیر lib/webservices به همین ترتیب جار فایل های مربوط به مولفه های وب سرویس را دارد.

۳. کد های جاوا و فایل های پیکر بندی محیط اجرا

این مسیر ها پرکاربرد ترین بخش شامل کد های جاوا و فایل های پیکر بندی می باشند که در این قسمت به توضیح آن ها می پردازیم. مسیر های src/web، src/java و src در این گروه می گنجند. جداول زیر به شرح فایل های موجود در این مسیر ها می پردازد.

نام فایل	توضیح
application.xml	فایل پیکربندی پکیج ear از روی این فایل و توسط مراحل ساخت ایجاد می شود.
log۴j.properties	در صورت استفاده از نرم افزار log۴j از این فایل پیکر بندی استفاده می شود.
opxiCallManager.properties	فایل پیکربندی کال منجر که در پکیج ear کپی می شود.
server-config.wsdd	فایل پیکربندی موتور Axis برای پیاده سازی وب سرویس ها
sip.xml	فایل پیکربندی پکیج sar، فایل web.xml اتوماتیک ساخته می شود
tiles-defs.xml, validator-rules.xml	فایل های پیکربندی چارچوب وب Struts

جدول ۲. لیست فایل های موجود در شاخه src

حال به شرح مختصری از ساختار کد های جاوا در مسیر های src/java و src/web می پردازیم. جدول زیر پکیج های موجود جاوا را تحت پکیج ریشه com.basamadco.opxi.callmanager به همراه وظایف آن ها نمایش می دهد.

نام پکیج	مسئولیت (مؤلفه ای که پیاده سازی می کند یا وظایف آن)
call	کلاس های مربوط به کنترل منطق مکالمات و تماس ها

crm	اتصال به سرویس ارتباط با مشتری برای دریافت آدرس وب خانگی مشتریان
entity	مولفه داده، موجودیت های داده ای کال منجر در این پکیج قرار دارند.
entity.dao	پیاده سازی الگوی DAO، واسط های آن در اینجا قرار دارند.
entity.dao.database	واسط های DAO برای دسترسی به موجودیت های پایگاه داده
entity.dao.database.hbm	پیاده سازی DAOی پایگاه داده با موتور Hibernate
entity.dao.directory	واسط های DAO برای دسترسی به موجودیت های دایرکتوری سرور
entity.dao.directory.ldap	پیاده سازی DAOی دایرکتوری سرور با LDAP
entity.dao.webdav	واسط های DAO برای دسترسی به موجودیت های سرور Webdav
entity.dao.webdav.exchange	پیاده سازی DAOی Webdav از طریق Exchange
logging	مؤلفه گزارشات
pool	مؤلفه استخر
profile	سرویس دسترسی به پروفایل ها
queue	مولفه صف (سرویس صف برای مکالمات)
util	این پکیج یک جعبه ابزار عمومی برای استفاده در کد کال منجر است.
sip	مرلفه سیپ (تمامی سیپ سرولت ها و کلاس های وابسته به آن ها)
sip.b۲bua	پیاده سازی ماشین های کنترل مکالمات
sip.front	سرولت های دریافت کننده پیام های اولیه سیپ
sip.listener	انواع سرولت های شنونده در کال منجر
sip.presence	سرولت های سرویس اعلان وضعیت
sip.proxy	سرولت های سرویس پراکسی
sip.queue	سرولت های سرویس صف
sip.registrar	سرولت سرویس رجیسترار
sip.security	سرویس امنیت برای تشخیص هویت کاربران در اتصال با کال منجر
sip.util	جعبه ابزار مورد استفاده پکیج سیپ
web	بخش وب کال منجر، شامل کنسول وب، وب سرویس ها و VXMLها

web.services	پیاده سازی سرویس های وب کال منجر
web.vxml	سرولت های مربوط به ایجاد فایل های VXML برای سرور رسانه

جدول ۳. شرح پکیج های موجود در کد جاوا

و در انتها به توضیح فایل های JSP می پردازیم. جدول زیر لیست فایل های JSP موجود در مسیر src/web را نمایش می دهد.

نام فایل	توضیح
template/	قالب کلی فایل های JSP و فایل های CSS مربوطه در این مسیرند.
AAL.jsp	صفحه نمایش گزارش کارکرد اپراتوری
application.jsp	صفحه نمایش کارکرد سرویس مدیا و استفاده از برنامه های صوتی
index.jsp	صفحه اصلی (خانگی) کنسول وب کال منجر
pool۲.jsp	صفحه نمایش وضعیت اپراتور های آنلاین
queue۲.jsp	صفحه نمایش صف های موجود در کال منجر
sendIM.jsp	صفحه ارسال پیام متنی فوری
show.jsp	صفحه نمایش جزییات سرویس صف
showCall.jsp	صفحه نمایش جزییات مکالمه
transfer.jsp	صفحه استفاده از سرویس انتقال مکالمه از طریق وب

جدول ۴. لیست فایل های موجود در شاخه src/web

۴. فایل های پیکربندی محیط توسعه

این مسیر ها شامل فایل های پیکربندی برای کامپایل و ساخت نرم افزار می باشد. مسیر ریشه این فایل ها src/resources نام دارد.
جدول زیر جزییات فایل ها و شاخه های موجود در این مسیر را به شما نشان می دهد.

نام فایل	توضیح

opxiActivityLog.xsd	شمای گزارشات کال منجر که برای تولید کد های مربوطه استفاده می شود.
opxiCMEntityProfile.xsd	شمای پروفایل های کال منجر که برای تولید کد های مربوطه استفاده می شود.
sip-app_۱_۰.dtd	فایل شمای استاندارد فایل sip.xml
wscompile-config.xml	فایل پیکربندی نگاشت پیشوند های XML و پکیج های جاوا برای تولید اتوماتیک فایل های پیکربندی وب سرویس ها.
hibernate/	فایل های پیکربندی Hibernate در این مسیر قرار دارند.
i۱۸n/	فایل های properties مربوط به پیغام های چند زبانه نرم افزار
sql/	فایل اسکریپت ایجاد شمای پایگاه داده (قابل ایجاد اتوماتیک)
xdoclet/	فایل های پیکربند ی XDoclet برا ی تولید فایل های پیکربندی بخش های وب

جدول ۵. لیست فایل های موجود در شاخه src/resources

۲.۲ ساختار اجرایی پروژه در زمان اجرا

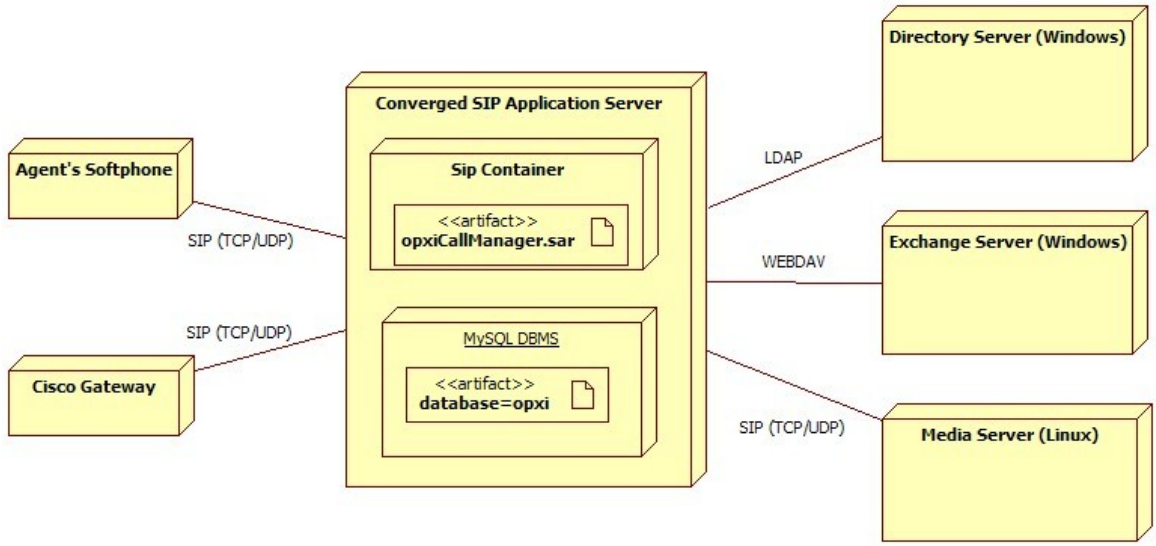
محیط عملیاتی نرم افزار کال منجر از یک سیپ سرور (مرکب با وب سرور) و یک سرور پایگاه داده تشکیل شده است. این سرور پایگاه داده همان داده های داخلی کال منجر را نگهداری می کند و به طور فیزیکی می تواند روی همان ماشین سرور سیپ قرار داشته باشد. کال منجر برای عملکرد صحیح به سرور مدیا، دایرکتوری سرور و سرور Webdav نیز احتیاج دارد. برای داشتن درک بهتری از محیط اجرایی پروژه و چگونگی ارتباط مؤلفه ها در زمان اجرا به شکل ۲ توجه کنید.

مکالمات ورودی، که از سمت شبکه های سوییچ تلفن معمولی می آیند، باید به سیگنالینگ سیپ تبدیل شوند تا برای کال منجر قابل استفاده باشند. دروازه[4] موجود در ساختار محیط عملیاتی همین نقش را دارد، یعنی برقراری ارتباط کال منجر و شبکه تلفن. کال منجر از سرور رسانه[5] برای پخش منو ها و برنامه های صوتی استفاده می کند. نقش سرور رسانه پخش فایل های صوتی و فایل های Voice XML بر اساس پروتکل سیپ است.

کال منجر برای مدیریت اپراتور ها، صف ها و هدایت صحیح مکالمات به اطلاعات پایه اپراتور ها، انواع برنامه های صوتی، صف ها و... اختیاج دارد. این اطلاعات روی سرور دایرکتوری ذخیره شده اند و کال منجر با اتصال به این سرور از طریق پروتکل LDAP پرس و جو های مورد نیاز را برای دریافت داده های مورد نظر انجام می دهد. اما کال

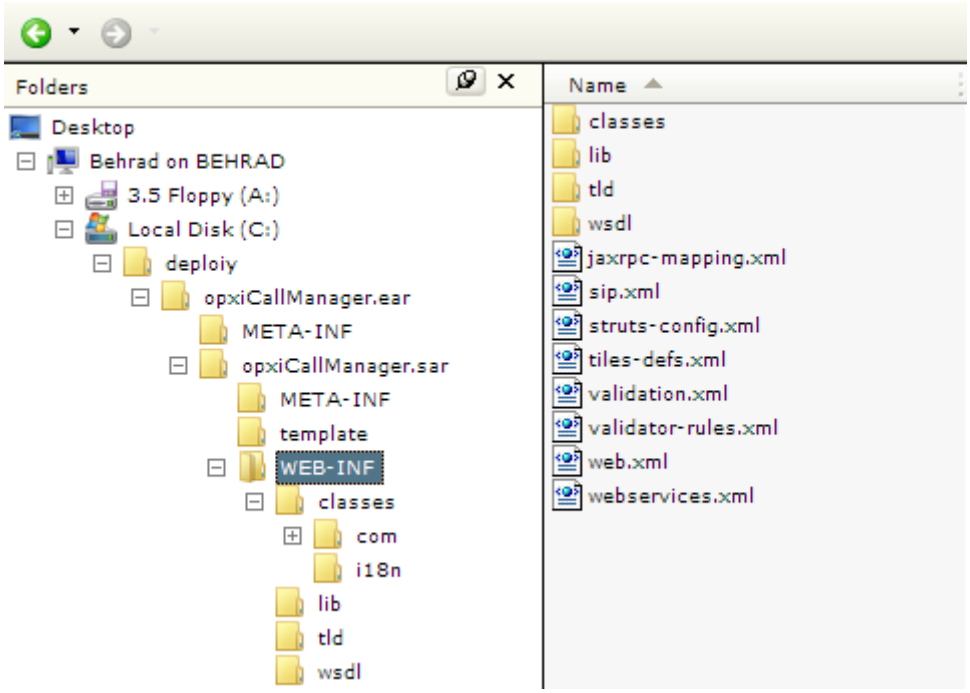
4 Gateway
5 Media Server

منجر نیاز به نگهداری اطلاعات خاصی برای خود دارد. چگونگی انتخاب و مدیریت صف ها، اجرای برنامه های صوتی، پیام های مختلف اپراتور ها و صف ها و ... از این قبیل اطلاعات هستند که در قالب پروفایل ها نگهداری می شوند. پروفایل ها و هر گونه اطلاعات دیگر مختص کال منجر برای موجودیت های دایرکتوری سرور، روی سرور اکسچنج و از طریق پروتکل Webdav انجام می شود.



شکل۲. شمای معماری فیزیکی در محیط اجرا

حال که دیدی سطح بالا از ساختار محیط عملیاتی کال منجر پیدا کردیم، زمان آن رسیده که به بررسی ساختار فایل اجرایی کال منجر بپردازیم. با توجه به این که کال منجر بر اساس Sip Servlet API v1.0 پیاده سازی شده است، باید به صورت یک فایل آرشیو در قالب SAR. روی سرور سیپ بارگزاری شود (با توجه به پشتیبانی سرور IBM Websphere از فایل های EAR که شامل فایل SAR باشند، ما نیز در حال حاضر نرم افزار را به صورت فایل EAR بارگزاری می کنیم). این فایل به گونه ای توسط مراحل کامپایل و ساخت تولید می شود که به سادگی روی سرور سیپ قابل نصب و بارگزاری است. ساختار این فایل کاملاً مطابق با استاندارد های J۲EE ۱.۴ و Sip Servlet API 1.0 است. فایل EAR، شامل فایل SAR است و فایل SAR مطابق استاندارد یک فایل مرکب سیپ و وب است که به تنهایی نیز می تواند روی سرور بارگزاری شود. ساختار فایل EAR را در شکل زیر می بینید.



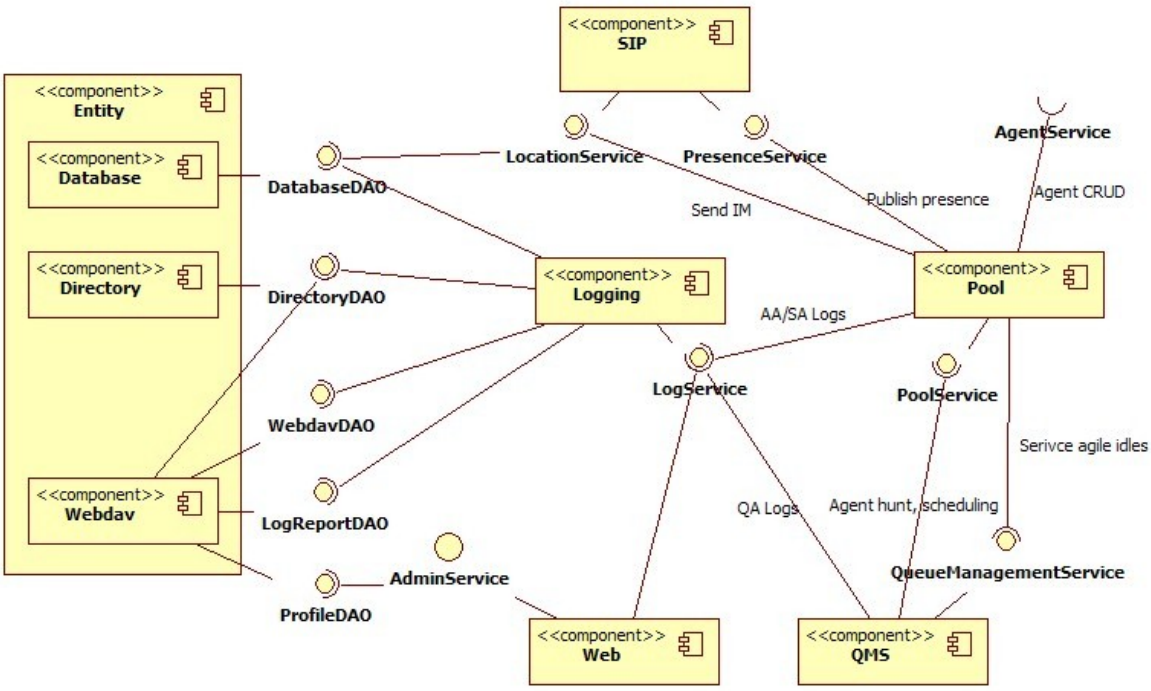
شکل۳. شمای ساختار فایل اجرایی ear

۳. معماری منطقی (طراحی مؤلفه ها)

نرم افزار کال منجر بر اساس سرویس های مورد نیازی که از او انتظار می رود از مولفه هایی تشکیل شده است که هر کدام از آن ها تمرکز خود را به پیاده سازی قسمتی از این سرویس ها معطوف کرده اند. نگاه ما به کال منجر در این بخش، بر اساس اینکه ساختار یک نرم افزار شی گرا را بررسی می کنیم، مبتنی بر مؤلفه خواهد بود. به طوری که ابتدا از بالا شمای کلی مولفه ها و واسط های آن ها را با هم می بینیم و بعد مؤلفه های موجود را تک تک بررسی می کنیم و البته سرویس های ارائه شده هر یک از مؤلفه ها را نیز خواهیم دید. برای آشنایی خاص تر با انواع سرویس های موجود در کال منجر نمودار کلاس سرویس ها نیز آورده شده است.

در نمودار زیر، مولفه های کلیدی موجود در کال منجر و واسط های ارائه شده توسط آن ها را می بینیم. روابط نیازمندی بین واسط ها نیز آورده شده و در مواردی روی این روابط توضیحاتی شامل نمونه سرویس مورد استفاده نیز آمده است.

در این قسمت برای درک و فهم بهتر نمودار مولفه ها و وابستگی های آنها، به تشریح مولفه ها (سرویس های) موجود در کال منجر می پردازیم. جزئیات این مولفه ها، ساختار درونی و نحوه عملیات آنها در ادامه مستندات آمده است.



شکل۴. مؤلفه ها و وابستگی های آن ها

۱. **مولفه داده**[6]: این مولفه مسئولیت مدیریت و نگهداری موجودیت های داده ای کال منجر را دارد. بسته به ماهیت موجودیت های داده ای، این مولفه به سه زیر مولفه تقسیم می شود.

۱.۱. **پایگاه داده**: وظیفه ذخیره و بازیابی داده های موجود در پایگاه داده داخلی کال منجر را برعهده دارد.

۲.۱. **دایرکتوری**: این مولفه برای خواندن و جستجوی داده های روی سرور دایرکتوری برای دسترسی به داده های اپراتورها، گروه ها، مهارت ها و برنامه های صوتی می باشد.

۳.۱. **وبدو**: این مولفه برای دسترسی به پروفایل های کال منجر روی سرور اکسچنج در نظر گرفته شده است.

۲. **مولفه سیپ**: یکی از اصلی ترین و بزرگترین مولفه های کال منجر می باشد که وظیفه اصلی آن دریافت پیام های سیپ و پاسخگویی به آنهاست. تمام تعاملات کال منجر در حیطه پروتکل سیپ از طریق این مولفه انجام می شود. سرویس های ثبت مکان، اعلان وضعیت اپراتورها، مدیریت مکالمات و .... از جمله سرویس های این مولفه هستند.

۳. **مولفه استخر**[7]: این مولفه مسئولیت مدیریت گروه های کاری و مهارت هایی که اپراتورها در آنها قرار دارند را برعهده دارد. این مولفه به کمک سرویس اپراتوری (AgentService) به اطلاعات

6 Entity Component
7 Pool Component

اپراتورها دسترسی دارد و سرویس PoolService را برای انتخاب اپراتور بهینه به مولفه مدیریت صف ارائه می دهد.

۴. **مولفه مدیریت صف**: در کال منجر صف ها مداخلی برای پاسخگویی اپراتورها به مکالمات ورودی هستند و از این رو پیچیدگی های خاصی برای نحوه انتخاب صف ها برای مکالمات ورودی، مدیریت مکالمات داخل صف ها و نهایتا اتصال مکالمه به اپراتور مورد هدف وجود دارد. مولفه مدیریت صف همین مسئولیت ها را دارد.

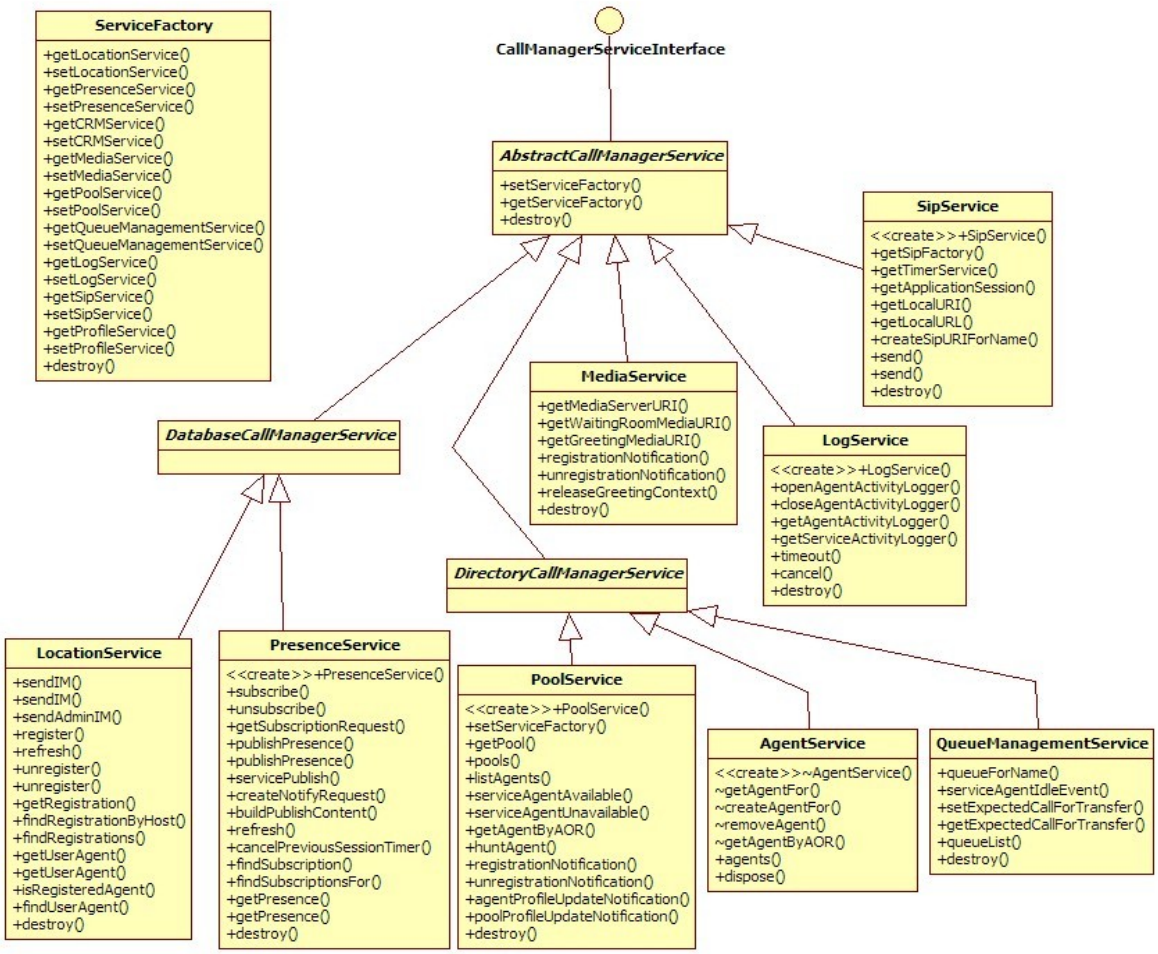
۵. **مولفه وب**: این مولفه کنسول مدیریت مبتنی بر وب کال منجر را دربر دارد . علاوه بر این وب سرویس های پیاده سازی شده در کال منجر در این مولفه تعبیه شده اند. وب سرویس مدیریتی AdminService از طریق سرویس ProfileService که در مولفه وبدو قرار دارد، کار می کند.

۶. **مولفه گزارشات**: این مولفه نیز وظیفه تهیه گزارشات کال منجر را بر اساس شمای تعریف شده گزارشات (در قالب XML) بر عهده دارد. سرویس های این مولفه و متدهای آنها امکان ثبت گزارش برای انواع رویدادهای کال منجر را در اختیار دیگر مولفه ها و سرویس هایشان قرار می دهند. در نهایت نگهداری، کنترل و نحوه ذخیره سازی گزارشات را همین مولفه برعهده می گیرد.

در ادامه نمودارهای کلاس سرویس ها را برای روشن تر شدن ساختار پیاده سازی سرویس ها آورده ایم.

۳.۱ نمودار کلاس سرویس ها

هر کدام از مولفه ها برای ارائه سرویس هایی بوجود آمده اند. هر سرویسی وابسته به یک یا چند تا از واسط های مولفه هاست. همه سرویس ها از کلاس پدر AbstractCallManagerService ارث می برند. این سرویس ها در ابتدای بار گزاری نرم افزار در محیط اجرایی، توسط کلاسCallManagerAppLoader راه اندازی می شوند تا در مخیط نرم افزار قابل استفاده باشند.
دسترسی به این سرویس ها از طریق کلاس ServiceFactory انجام می شود . این کار با فراخوانی متد getServiceFactory چه در داخل سیپ سرولت ها و چه داخل خود سرویس ها امکان پذیر است.



شکل ۵. نمودار کلاس سرویس ها

۳.۲ واسط های خارجی

نرم افزار کال منجر سرویس هایی را به دنیای بیرون خود ارائه می دهد. مشتریان این سرویس ها هم کاربران آن هستند و هم نرم افزارهای دیگر مجموعه OPXI، در این بخش سعی داریم واسط های موجود ارائه شده توسط این سرویس ها را شناسایی کنیم:

۱. **واسط سیپ:** این شاید اصلی ترین و پرکاربرد ترین سرویس نرم افزار می باشد که به وسیله آن و از طریق پروتکل SIP ، کاربران می توانند به صورت استاندارد با کال منجر تعامل داشته باشند . انواع سرویس های موجود SIP در کال منجر را در بخش مولفه سیپ بررسی خواهیم کرد. توجه داریم که مولفه سیپ در کال منیجر به طور عمده وظیفه پیاده سازی و پاسخگویی به این سرویس ها را دارد. مشتریان این سرویس هم کاربران کال منجر هستند هم نرم افزارهای دیگر مجموعه OPXI.

۲. **واسط وب:** کال منجر یک کنسول مدیریتی مبتنی بر پروتکل HTTP دارد که از طریق آن می توان برخی عملیات کنترل و نظارت و یا مدیریتی را انجام داد. جزئیات بیشتر این بخش را در قسمت مربوط به مولفه وب توضیح خواهیم داد. مشتریان این سرویس کاربران و مدیران کال منجر هستند.

۳. **وب سرویس ها:** آخرین مدخل خارجی برای تعامل با کال منجر وب سرویس های آن هستند. نرم افزار کال منجر برای تعامل با دیگر نرم افزارهای موجود در OPXI و عمدتا به منظور انجام عملیات مدیریتی و پیکربندی، سرویس های وبی را بر اساس پروتکل SOAP و استانداردهای وب سرویس مبتنی بر J2EE ۱.۴ پیاده سازی کرده است.

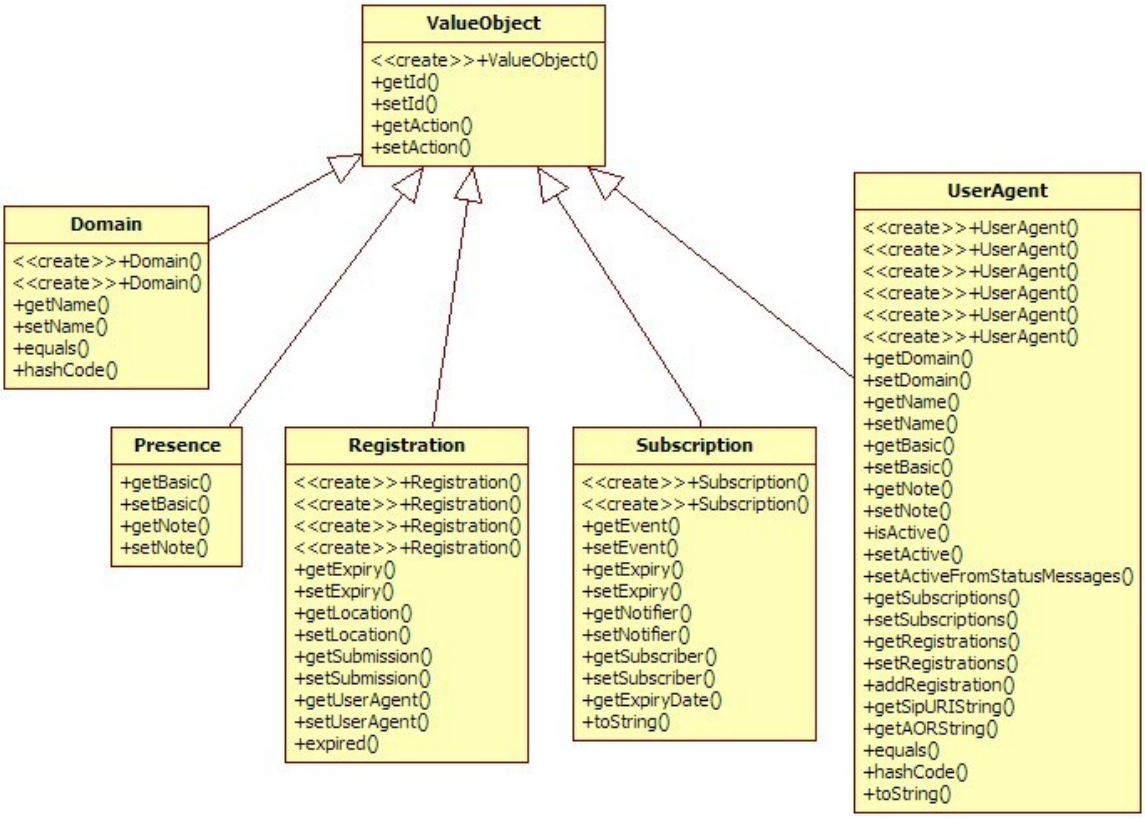
حال به بررسی جزییات مولفه های کال منجر می پردازیم.

۴. مؤلفه داده (Entity)

این مولفه موظف است موجودیت هایی که داخل سیستم بیشتر جنبه داده ای دارند را مدیریت کند، ذخیره و بازیابی انواع موجودیت های داده ای اصلی ترین سرویس این مولفه است. موجودیت های داده ای در کال منجر شامل داده های پایگاه داده داخلی نرم افزار برای ارائه سرویس های SIP شامل رجیستری و اعلان وضعیت، اطلاعات ساختاری اپراتور ها، گروه های کاری، مهارت ها و برنامه های صوتی در داخل سرور دایرکتوری و نهایتاً انواع گزارشات و پروفایل ها در سرور Exchange، می باشند. جزییات هرکدام از موارد گفته شده را خواهیم دید. قابل ذکر است طراحی این مولفه به صورت الگوی DAO انجام گرفته است [۶]. این مولفه خود به سه زیرمولفه دیگر تقسیم می شود.

۴.۱ Database

مسئول مدیریت موجودیت هایی که در پایگاه داده داخلی کال منجر،که هم اکنون MySQL است، نگهداری می شوند. سرویس هایی که نیاز به دسترسی به پایگاه داده داخلی کال منجر را دارند باید از سرویس DatabaseCallManagerService که برای این منظور در نظر گرفته شده است ارث بری کنند.



شکل۶. نمودار کلاس موجودیت های پایگاه داده

پیاده سازی: دسترسی به پایگاه داده و مدیریت کنترل و دستکاری جدول ها از طریق نرم افزار Hibernate انجام می شود. برای این کار کلاس های داده ای توسط تگ های XDoclet که در قالب توضیحات Javadoc می باشند خود را به تسک مربوط به ساخت متاداده های مورد نیاز Hibernate در مرحله ساخت[8]، معرفی می کنند. این تگ ها همه بصورت "@hibernate" شروع می شوند و از داخل کد کلاس های داده ای در شاخه database قابل دسترسی هستند. نگاشت کلاس های داده به جداول پایگاه داده از طریق همین متا داده ها و توسط Hibernate انجام می شود. کدهای مربوط به استفاده از Hibernate برای دسترسی به داده های پایگاه داده (در قالب الگوی DAO) در شاخه (پکیج) hbm یافت می شوند.

8 Build

Directory ۴.۲

مسئول خواندن اطلاعات موجودیت هایی که روی سرور LDAP قرار دارند می باشد. نرم افزار کال منجر به اطلاعات اپراتور ها، مهارت ها، گروه ها، برنامه های صوتی و... برای عملکرد خود احتیاج دارد. سرویس DirectoryCallManagerService برای این منظور (دسترسی به اطلاعات دایرکتوری سرور) در نظر گرفته شده است.

پیاده سازی: دسترسی به سرور LDAP از طریق پکیج ldap انجام می شود. کلاس های این پکیج نیز مطابق با الگوی DAO ایجاد شده اند. کلاس پدر این پکیج LdapDAO می باشد. کال منجر تمام اطلاعات ساختاری مربوط به اپراتور ها، گروه ها و مهارت های کاری و برنامه های صوتی را از داخل سرور دایرکتوری می خواند اما اطلاعات رفتاری این موجودیت ها از طریق فایل های XMLی که پروفایل نام دارند و در سرور Exchange ذخیره می شوند، خوانده می شود. این کار از طریق سرویس پروفایل (کلاس ProfileService) و مؤلفه دسترسی به سرور Exchange انجام می شود.

۴.۲.۱ کلاس CallTarget

موجودیت هایی که اطلاعات آن ها از روی دایرکتوری سرور خوانده می شوند، موجودیت هایی هستند که به عنوان مقصد و هدف مکالمات (مثل اپراتورها، صف ها، گروه های کاری و یا برنامه های صوتی) استفاده می شوند. از این رو این مجموعه کلاس ها به عنوان فرزندان کلاسی با نام CallTarget در نظر گرفته شده اند که نمودار کلاس آن ها را در زیر می بینید.

همانطور که مشخص است، هر هدف مکالمه علاوه بر خصوصیات پایه از قبیل شماره تلفن و نام یک پروفایل برای تعریف ویژگی های مد نظر برای کال منجر دارد. این پروفایل ها از طریق پروتکل WEBDAV روی سرور Exchange نگهداری می شوند و شمای آن ها نیز در فایل OpxiCMEntityProfile.xsd تعریف می شوند. از همین رو هر زیر نوع از کلاس CallTarget می بایست مواردی را در پیاده سازی خود رعایت کند که در اینجا به این موارد اشاره می کنیم.

۱. **پیاده سازی عمل خواندن از روی سرور دایرکتوری:** برای خواندن خصوصیات و فیلدهای موجود در دایرکتوری باید از طریق الگوی DAO و مانند کلاس هایی نظیر LdapAgentDAO و یا LdapCallTargetDAO شی هدف مکالمه مورد نظر به درستی مقدار دهی شود.

۲. **پیاده سازی متد های انتزاعی:** متد های انتزاعی کلاس CallTarget باید پیاده سازی شوند. این متد ها چنینند:

```
public abstract URI getTargetURI() throws OpxiException;
```

این متد آدرسی (SIP URIیی) را تولید کرده و برمی گرداند که کال منجر باید مکالمه را به آن آدرس هدایت کند.

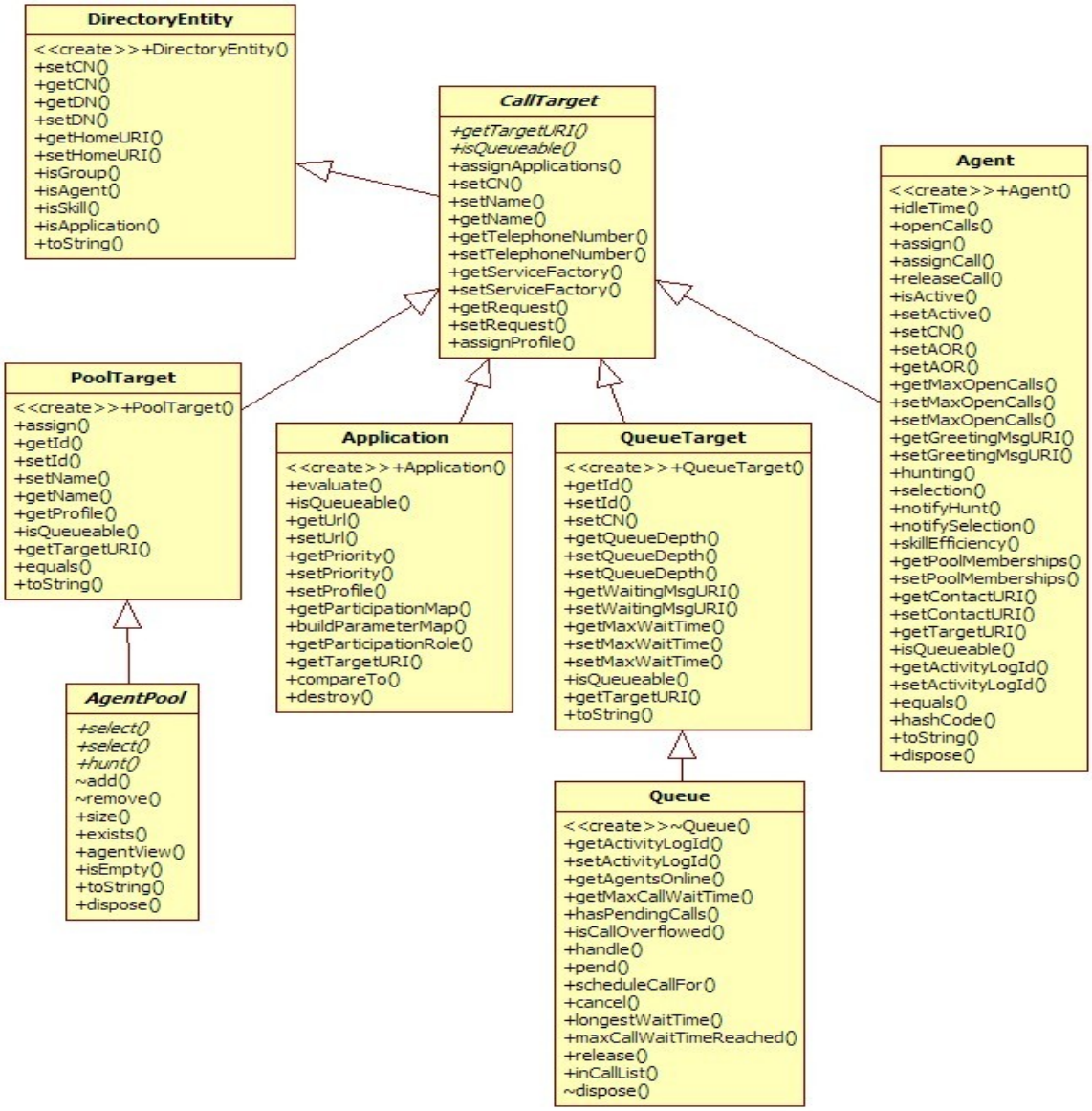
```
public abstract boolean isQueueable();
```

نوع سرویس دهی (صف یا پراکسی) برای اتصال مکالمه به این هدف را مشخص می کند.

```
protected abstract boolean hasUpdatableState();
```

شرایطی که تغییرات ایجاد شده در پروفایل این شی هدف مکالمه، قابل اعمال روی شی مقیم در حافظه است.

۳. تغییر پیاده سازی متد applyProfile: این متد مقادیر موجود در پروفایل را روی شی هدف اعمال می کند. پیاده سازی موجود در کلاس CallTarget فقط قسمت مشترک پروفایل ها یعنی برنامه های صوتی را اعمال می کند. پس هر زیر نوع باید با پیاده سازی مجدد این متد، تنظیمات مورد نظر خود را از پروفایلش به خصوصیات خود اعمال کند. برای مثال به پیاده سازی این متد در کلاس Agent توجه کنید.



شکل ۷. نمودار کلاس موجودیت های هدف تماس

Webdav ۴.۳

مسئول خواندن و نوشتن اطلاعات روی سرور Exchange با استفاده از پرو تکل Webdav می باشد. این اطلاعات شامل گزارشات (LogService)و پروفایل های (ProfileService) کال منجر می شود.

پیاده سازی: پیاده سازی این مؤلفه مطابق با الگوی DAO در پکیج exchange موجود می باشد.

۵. مؤلفه Pool (مدیریت اپراتور ها، مهارت ها و گروه های کاری)

سرویس های مدیریت اپراتورها، گروه ها و مهارت ها و انواع الگوریتم های انتخاب اپراتور بهینه برای پاسخ گویی به تماس ها را ارائه می دهد. در کال منجر مجموعه اپراتور های متصل به سیستم پاسخ گویی (با استفاده از سرویس اعلان وضعیت[9]) توسط "*استخر هایی*[10]" از اپراتور ها نگهداری و کنترل می شوند. به این ترتیب که این سرویس بعد از دریافت پیام از سرویس سیپ اعلان وضعیت و اطلاع از حضور، تغییر وضعیت و یا خروج اپراتور از سیستم با رجوع به گروه های کاری و مهارت هایی که اپراتور مذکور در آنها عضویت دارد، وضعیت آن ها را به روز می کند. مجموعه های اپراتوری یا همان "استخرها" خود بر دو نوع هستند: *گروه های کاری*[11] و *مهارت ها*[12] (نمودار ۶ را در ادامه ببینید). الگوریتم های انتخاب بهترین اپراتور آزاد بر اساس مهارت ها و گروه های کاری به دو گروه تقسیم می شوند چون مهارت ها الگوریتم های پیچیده تری را شامل می شوند در حالی که الگوریتم های انتخاب اپراتور از گروه های کاری ساده تر می باشند..

۵.۱ انتخاب اپراتور بهینه مبتنی بر مهارت

ساده ترین الگوریتم برای انتخاب اپراتورها، انتخاب بر اساس طولانی ترین زمان استراحت[13] است. در این الگوریتم اپراتوری که ارتباطش زودتر (قبل) از دیگران تمام شده باشد، برای پاسخ دهی به مکالمه ورودی به سیستم صف انتخاب می شود. اما زمانی که بحث مهارت های مختلف کاری مطرح می شود، باید از بین اپراتور ها، به گونه ای انتخاب کرد که هم مدت کل زمان اشغال بودن اپراتور ها نرمال باشد ، هم فرد انتخاب شده از نظر دارا بودن مهارت های لازم، بهترین فرد باشد. برای این منظور در کال منجر یک الگوریتم سه مرحله ای در نظر گرفته شده است (کلاس PoolService را ببینید) که به این ترتیب عمل می کند:

۱. ابتدا استخر اولیه ای برای پاسخ گویی به تماس موردنظر هم نام با نام صف تشخیص داده شده برای آن تماس انتخاب می شود و متد select آن جهت امتیاز دهی اولیه اپراتور های موجود در آن استخر بر اساس مهارت هایشان و اولویت پاسخ دهی، فراخوانی می شود و امتیازات اپراتور ها توسط کلاس AgentSelection بر می گردد.

۲. استخر های بار گزاری شده در کال منجر تک تک برای امکان اعمال روی تماس چک می شوند. این چک توسط الگوریتم هایی بنام Matching Rules انجام می شوند. اگر پاسخ استخری بلی بود، متد select آن همراه پارامتر امتیازات کنونی اپراتور ها فراخوانی می شود. این متد به اپراتور هایی که بین این استخر و استخر اولیه مشترک هستند امتیاز می دهد.

---

9 Presence Service
10 Pool
11 Workgroup
12 Skill
13 LIAR agent hunting algorithm

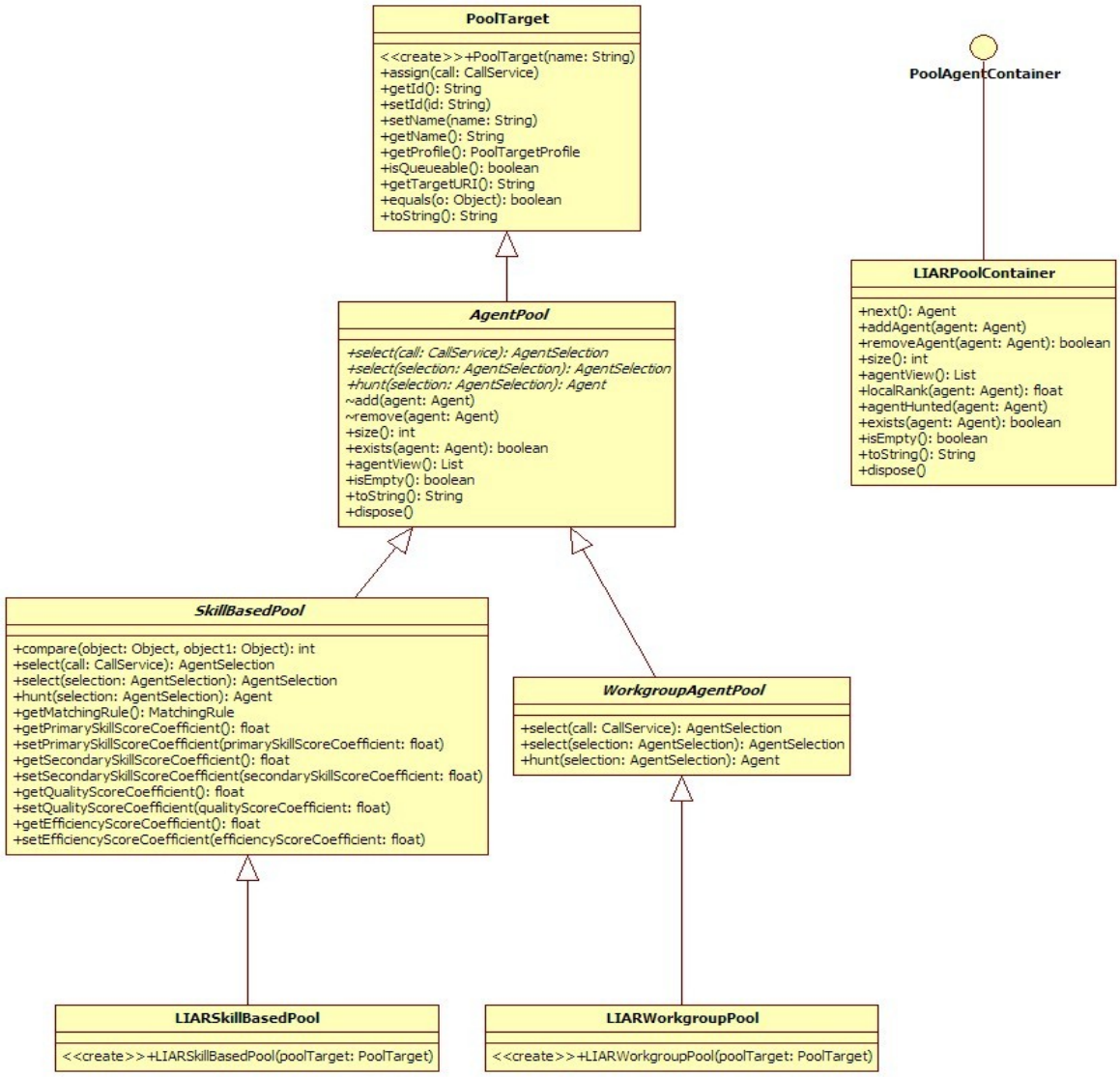
۳. پس از بررسی تمامی استخر ها، استخر اولیه اقدام به انتخاب[14] اپراتور با بیشترین امتیاز می کند. برای این منظور ابتدا نتایج نهایی را بر اساس الگوریتم مرتب سازی خود مرتب می کند و بعد از اولین نفر شروع به انتخاب می کند. اگر انتحاب اپراتور به دلایلی ممکن نباشد یا مشکلی پیش بیاید، نفر بعدی انتخاب می شود و ...

برای پیاده سازی و امتیاز دهی به اپراتور ها، میزان مهارت هر اپراتور با تعریف ضریبی برای عضویت او در آن مهارت مشخص می شود. هر مهارت نیز دو ضریب جداگانه، یکی به عنوان ضریب مهارت اولیه و دیگری ضریب مهارت ثانویه، دارد که اولی برای امتیاز دهی به اپراتور ها وقتی که مهارت به عنوان استخر اولیه انتخاب شده (در مرحله اول) و دومی برای امتیازدهی توسط استخر های ثانویه (در مرحله دوم) به کار می رود.

الگوریتم های مرتب سازی مهارت ها، امتیاز مهارتی اپراتور ها را با رتبه آن ها بر اساس الگوریتم LIAR که قبلا گفته شد تلفیق می کند و بعد اپراتور ها را یر اساس مجموع امتیاز آن ها مرتب می کند. تلفیق امتیاز ها بر اساس دو ضریب کیفیت[15] و کارایی[16] که می توانند برای هر استخر جداگانه تعریف شوند، انجام می شود. بیشتر بودن ضریب کیفیت نسبت به ضریب کارایی بهتر بودن میزان مهارت انتخاب شده را نشان می دهد و بیشتر بودن ضریب کارایی نسبت به ضریب کیفیت عادلانه تربودن انتخاب بر اساس الگوریتم کارایی اپراتور ها (LIAR) را نتیجه می دهد.

نمودار کلاس استخر ها را در زیر می بینید.

14 Hunt
15 QualityScoreCoefficient
16 EfficiencyScoreCoefficient



شکل ۸. نمودار کلاس استخر ها

۶. مؤلفه سیپ

از آنجایی که کال منجر به عنوان یک نرم افزار ارتباطی مبتنی بر SIP طراحی و پیاده سازی شده است، این مولفه هسته ای ترین مولفه کال منجر است که وظیفه برقراری، کنترل و مدیریت تماس ها و بطور کلی سرویس های مبتنی بر پروتکل SIP را عهده دار است. به طور کلی می توان گفت تمام رویداد های ورودی این نرم افزار پیغام های SIPی هستند که از لایه سیپ-سرولت به نرم افزار وارد شده و برای گرفتن سرویس خود از آن طریق در اختیار مولفه های دیگر قرار می گیرند. به علت همین طراحی مولفه گرا می توان گفت اکثر مولفه های کال منجر (با اندک تغییراتی) در محیط های دیگر غیر از SIP نیز قابل استفاده مجدد هستند.

۶.۱ چارچوب پکیج سیپ

این مولفه در اصل از سیپ سرولت ها تشکیل شده است. سیپ سرولت ها وظیفه دریافت و ارسال پیام های سیپ را مطابق استاندارد Sip Servlet API v.1.0 دارند. در کال منجر سرویس های مختلفی مبتنی بر پروتکل سیپ پیاده سازی شده است. هر کدام از این سرویس ها توسط یک یا چند سیپ سرولت مرتفع می شوند. این سیپ سرولت ها در زیر پکیج مربوطه داخل پکیج سیپ قرار دارند. قبل از اینکه وارد جزئیات هر یک از این سرویس ها شویم، سرولت های مهم پکیج سیپ، که چارچوب ابن پکیج را می سازند را بررسی می کنیم.

OpxiSipServlet ۶.۱.۱

این سرولت هسته چارچوب کاری سرولت های دیگر کال منجر می باشد و در واقع گسترشی بر سیپ سرولتهای استاندارد است. امکاناتی که این سرولت به مجموعه سرولت های کال منجر می افزاید چنین است.

۱. مجموعه متدهای کنترلی غنی تر:

doPublish، doRefer، doInitialInvite، doReInvite

۲. برقراری انحصار دو جانبه در فراخوانی بین متدهای doRequest و doResponse در یک سشن سیپ

۳. امکان بررسی هویت کاربری[17] هنگام دریافت پیام های سیپ: هر سرولتی که بخواهد از این امکان امنیتی استفاده کند باید متد toValidate را پیاده سازی کند و مقدار True را در مواقعی که تشخیص هویت در پیام سیپ باید فعال باشد ، برگردا ند (به عنوان مثال به سرولت PresenceServlet توجه کنید).

۴. دسترسی به مجموعه سرویس های کال منجر از طریق متد getServiceFactory

۶.۱.۲ سیپ سرولت های شنوده

سرولت های شنونده، سرولت هایی هستند که با معرفی در فایل sip.xml با اتفاق افتادن رویدادهای خاصی، توسط سیپ سرورفراخوانی می شوند. مولفه سیپ کال منجر در حال حاضر ۳ سرولت شنونده را تعریف کرده است.

17 Authentication

۱. ApplicationSessionManager: این سرولت مدیریت طول عمر SipApplicationSession ها را برعهده دارد.

۲. AckNotReceived: این سرولت پیام های خطا شامل عدم دریافت پیام Ack را سرویس می دهد.

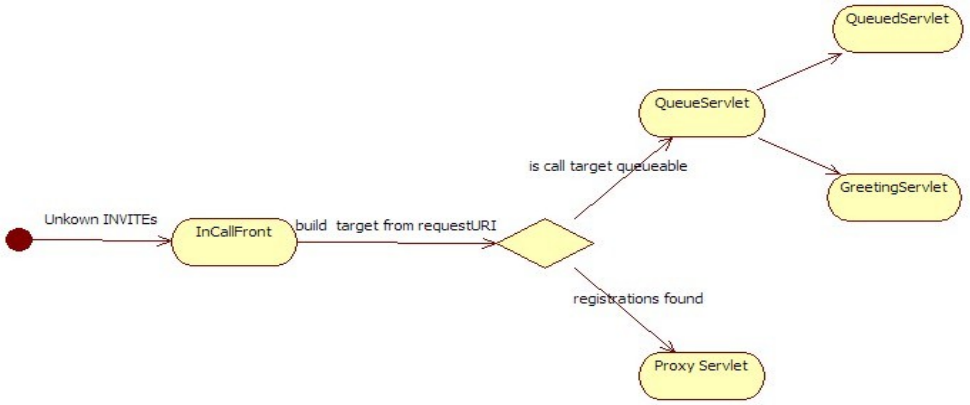
۳. CallManagerTimerListener: این سرولت وظیفه فراخوانی متدهای timeout تایمرهای کال منجر را برعهده دارد. تایمر ها در کال منجر همگی از کلاس AbstractTimerContext ارث می برند. برای افزودن تایمر جدید لازم است بعد از تعریف یک کلاس فرزند جدید برای آن تایمر، متدهای constructor و timeout را مطابق با نیازهای خود پیاده سازی کنید. به عنوان مثال می توانید به کلاس PresenceTimer رجوع کنید. {تایمر های موجود تشریح شوند}

۶.۱.۳ سیپ سرولت های فراخوانی شونده با پیام های سیپ اولیه

گروه دوم سیپ سرولت هایی هستند که برای پاسخ گویی به پیام های اولیه سیپ ورودی، با رسیدن پیام، توسط سرور سیپ فراخوانی می شوند. این سیپ سرولت ها در فایل sip.xml به عنوان سرولت های گیرنده پیام های اولیه با استفاده از فیلتر های Sip Servlet API معرفی شده اند.

۱. InCallFront: دریافت کننده تمامی پیام های Invite های اولیه که هم پارامتر outboundLeg را ندارند هم جز موارد ۲ و ۳ نیز نبوده اند. این پیام ها شامل تمامی INVITEها بجز موارد فیلتر شده توسط GreetingFront و PostGreetingFront می باشند که در این سرولت پردازش می شوند و بعد از تشخیص داده شدن هویت موجودیت مورد هدف پیام از روی RequestURI به یکی از سرولت های ProxyServlet، جهت پراکسی به سمت مقصد و یا QueueServlet، جهت ارائه سرویس صف (برای اتصال به اپراتور)، هدایت[18] می شود. نمودار زیر این روند را نمایش می دهد. سرولت QueueServlet بسته به اینکه اپراتور آزاد برای پاسخ دهی وجود داشته باشد پیام را به GreetingServlet و در غیر این صورت، جهت انتظار به سرولت QueuedServlet می فرستد.

18 Forward



شکل ۹. نمودار روندچرخش پیام های Invite اولیه نا شناخته

۲. GreetingFront: دریافت کننده تمامی پیام های Inviteای که بخش نام کاربری آن ها *"۸۸۶۷۸۷۶۸۲۶"* باشد. این پیام ها، پیام هایی هستند که از طرف Cisco Gateway در مواقع *انتقال مکالمات* به اپراتور (جهت پخش پیام خوش آمدگویی اپراتور) تعریف شده اند و فرستاده می شوند.

شکل ۱۰. نمودار روندچرخش پیام های Invite در انتقال مکالمه به اپراتور

۳. PostGreetingFront: دریافت کننده تمامی پیام های Inviteای می باشد که بخش نام کاربری آنها *"postGreeting"* باشد. این پیام ها، پیام هایی هستند که از طرف سرور رسانه ای برای انتقال مکالمه به اپراتور پس از پخش پیام خوش آمدگویی به کال منجر فرستاده می شوند.

شکل ۱۱. نمودار روندچرخش پیام های Invite در اتصال به اپراتور

۴. RegistrarServlet: دریافت کننده پیام های Register جهت سرویس رجیسترار[19] می باشد.

19 Location Service

۵. PresenceServlet : دریافت کننده پیام های Subscribe، Publish و Notify جهت سرویس اعلان وضعیت می باشد.

۶. IMServlet: پیام های Message را جهت پیاده سازی سرویس پیام متنی فوری[20] دریافت می کند.

۷. ConfigRobot: این سرولت پیام های Messageی را دریافت می کند که به نام کاربری *"opxiCallManager"* فرستاده شده باشند. این پیام ها واسطی جهت کنترل و مدیریت کال منجر از طریق پیام های متنی سیپ درنظر گرفته شده اند.

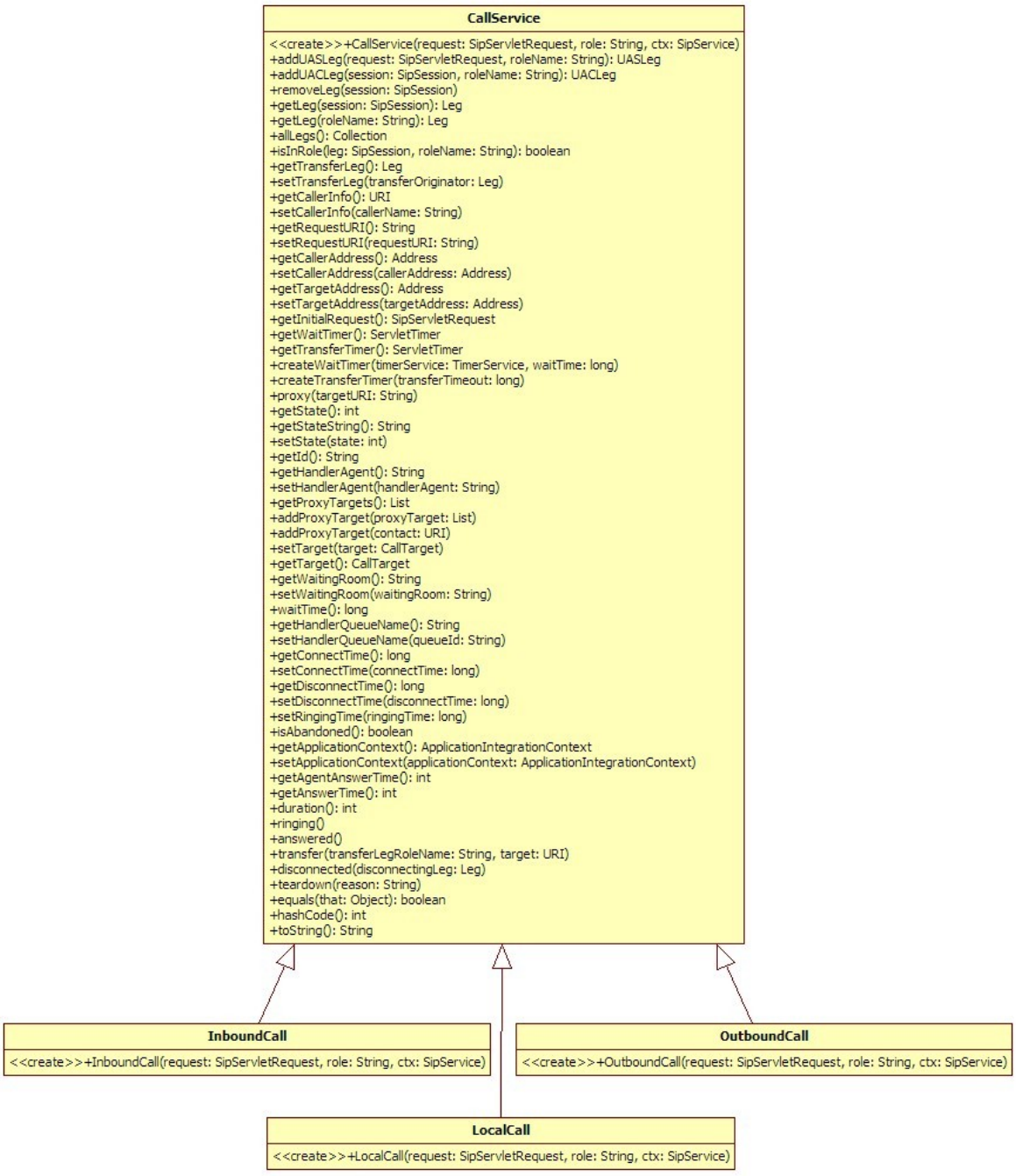
۶.۲ لایه مدیریت مکالمات

کنترل مکالمات یکی از عمده ترین وظایف کال منجر می باشد. برای این منظور کلاس هایی از روی دامنه مسئله تشخیص داده شده اند که وظیفه نمایش اطلاعات و اجرای منطق لازم برای تماس ها را داشته باشند. این کلاس ها پکیج call را تشکیل می دهند و در واقع هسته کلاس های لایه منطق نرم افزار کال منجر هستند. طراحی این کلاس ها در ابتدای امر بدون وابستگی به پروتکل سیپ انجام گرفته تا این لایه مستقل از جزئیات این پروتکل (SIP Servlet API) بتواند منطق لازم را برآورده سازد. اما در روند توسعه کال منجر (با توجه به این که این موضوع به عنوان یک معیار در درجه اول مطرح نبوده است) این استقلال کمی کمرنگ شده است. این موضوع را با مشاهده کلاس های داخل پکیج call و وابستگی های آنها به SIP Servlet API در برخی موارد، می توان به سادگی فهمید.

۶.۲.۱ کلاس CallService

اصلی ترین کلاس در این پکیج، کلاس CallService می باشد که نماینده حضور یک تماس الکترونیکی می باشد. به این ترتیب با آمدن تماس جدید، یک شی از یکی از فرزندان این کلاس (بر اساس نوع تماس وارده) ایجاد می شود و بعد از دریافت اطلاعات لازم از روی دایرکتوری سرور و مشخص شدن هدف شی تماس، این شی متناسب با نوع هدف، سرویس داده می شود. چگونگی تشخیص هدف تماس از روی پیام INVITE وارده در قسمت بعدی توضیح داده شده است. نمودار کلاس مربوطه را در زیر می بینید.

20 Instant Message



شکل ۱۲. نمودار کلاس CallService

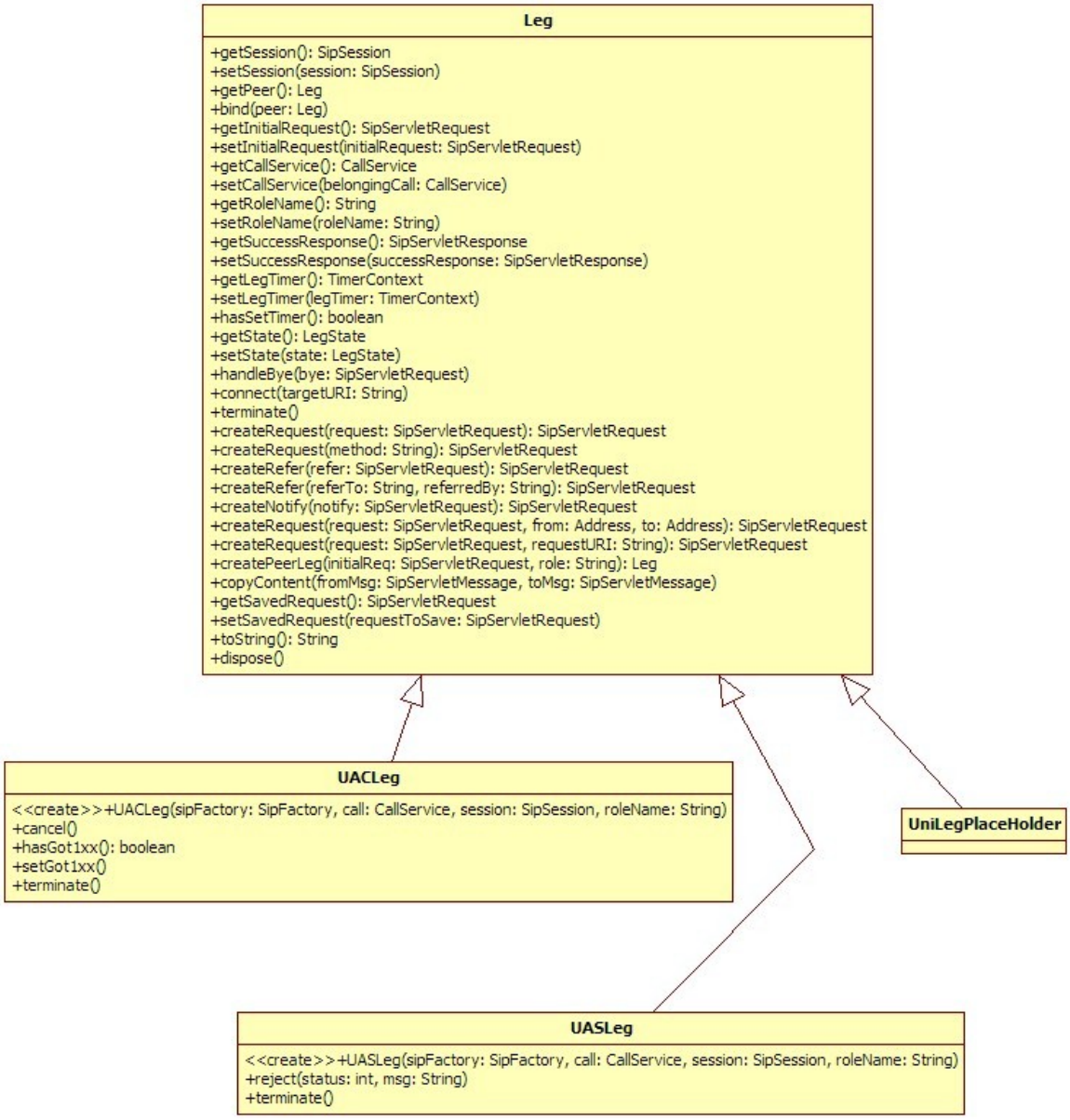
۶.۲.۲ کلاس Leg

کلاس مهم دیگر، کلاس Leg می باشد. این کلاس معرف یک طرف درگیر در هر مکالمه است. اصولاً هر مکالمه از تعدادی طرف یا *لگ*[21] تشکیل می شود که این لگ ها ارتباط مستقیم با سشن[22] های پروتکل SIP که طرف های مکالمه را تشکیل می دهند دارند. هر لگ موظف است پیغام ها و رویداد های SIP مربوط به سشن خود را سرویس دهد و در حالت کلی کنترل مکالمات از طریق کنترل تک تک لگ هایش انجام می شود (زمانی که تمام لگ های یک شی مکالمه از بین رفته باشند، خود شی مکالمه نیز از حافظه جمع می شود)، یعنی اگر یک مکالمه کنفرانس شامل ۳ طرف باشد، شیء CallService مربوطه داخل خود ۳ شی Leg خواهد داشت که هر کدام نماینده یکی از طرف ها می باشند. در مدل پروتکل های ارتباطی مثل SIP طرف های مکالمه به دو نوع تقسیم می شوند. یکی عامل فراخواننده (UAC) و دیگری عامل فراخوانده شده (UAS). واضح است که برای طرفین ورودی (عامل فراخواننده کال منجر) از UAS و برای طرفین خروجی (که کال منجر آن ها را فرا می خواند) از UAC استفاده می شود که از دیدگاه پروتکل سیپ، هر کدام منطق خاص خود را برای کنترل سیگنالینگ دارند.

با رسیدن یک پیام Invite اولیه، یک شی از کلاس CallService ایجاد می شود و اطلاعات لازم در آن پر می شوند. سپس برای نمایش طرف فراخواننده مکالمه یک شی Leg (لگ) ایجاد شده و در کلاس CallService قرار می گیرد. لگ ها در مراحل مختلف با دریافت رویدادهای مختلف تغییر وضعیت می دهد که این تغییر وضعیت ها در نمودارهای حالت هر یک از سرویس های مکالمه آورده شده است. شی تماس در انتها پس از بین رفتن تمامی طرف های درگیر (لگ های خود)، از بین می رود.

---

21 Leg
22 Session



شکل ۱۳. نمودار کلاس لگ ها

۶.۳ تشخیص هدف مکالمه

برای تشخ یص هد ف یک تماس ورودی، ابتدا ب خش username آدرس سیپ دا خل ق سمت RequestURI پیام Invite اولیه به عنوان کلید جستجو انتخاب می شود. سپس با جستجو در فیلد شماره تلفن (اگر username یک شماره بود) و یا فیلد نام موجودیت های موجود در سرور دایرکتوری (تحت همان مسیر ریشه ای که برای دایرکتوری سرور در فایل پیکربندی کال منجر تعریف شده است) موجودیت هدف را پیدا می کند. اگر چنین موجودیتی پیدا

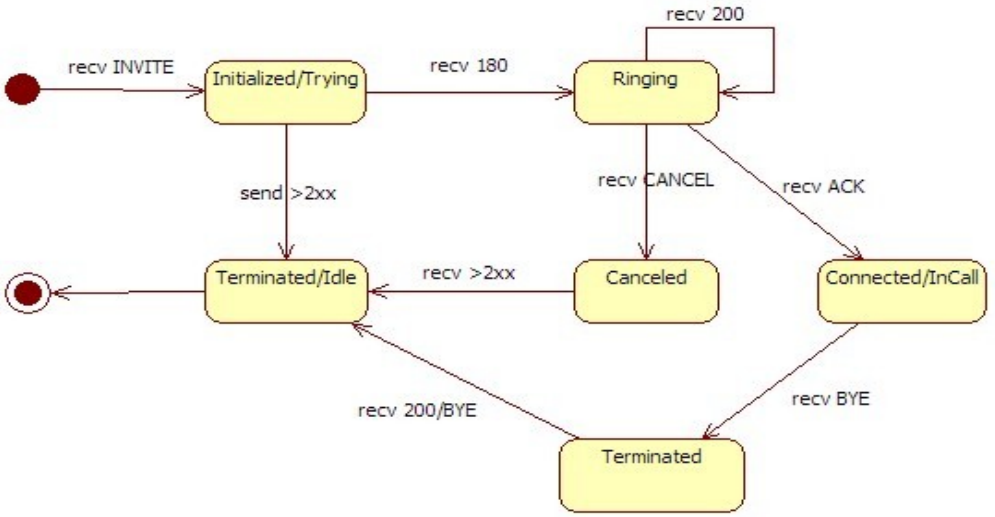
شد، یک شی CallTarget (به نمودار کلاس CallTarget توجه کنید) از نوع مناسب ایجاد شده و به شیء تماس منتسب می شود. سپس با استفاده از سرویس رجیسترار (LocationService) آدرس (های) فیزیکی موجود در کال منجر برای این هدف، شناسایی می شوند. اگر آدرس فیزیکی ای در دسترس نبود، پاسخ با کد ۴۰۴ برمی گردد و وضعیت طرف (لگ) فراخواننده به Idle تغییر می کند. در حالت عادی و درصورت پیدا شدن آدرس فیزیکی (بسته به نوع شی هدف)، از سرویس های صف و یا پراکسی برای سرویس دهی به تماس ورودی استفاده می شود. توجه داریم که اگر موجودیتی در سرور دایرکتوری با کلید مورد نظر موجود نباشد نیز، پاسخ ۴۰۴ برمی گردد.

۶.۴ سرویس های پاسخ دهی به مکالمات

در کال منجر برای سرویس دهی به تماس ها بعد از تعیین شدن شی هدف تماس، دو مدل سرویس در نظر گرفته شده است: سرویس صف و سرویس پراکسی. سرویس پراکسی برای اتصال فراخوان به منوها و برنامه های صوتی (سرور رسانه ای) و اپراتورها به یکدیگر (تماس های داخلی) و سرویس صف برای اتصال فراخوان به اپراتور در تماس های ورودی از دنیای خارج محیط عملیاتی استفاده می شوند.
پیاده سازی سرویس پراکسی، از نظر منطق سیگنالینگ، به مراتب ساده تر بوده و تغییر وضعیت شیء تماس محدود تر است ولی در سرویس صف برای پیاده سازی سناریوهای مختلف کنترل و سیگنالینگ مکالمه نیاز به ماشین های حالت b2bua مختلف و در مواردی پیچیده خواهیم داشت. پیاده سازی این ماشین ها در حال حاضر در دو بخش، انجام شده است. بخشی در کلاس لگ و بخش دیگر در سیپ سرولت های B2BUA و فرزندانش داخل پکیج b2bua. این سرولت ها با دریافت پیام های SIP، وضعیت طرف ها و در نتیجه شی تماس را به روز می کنند.

۶.۴.۱ سرویس پراکسی

این سرویس، که ساده ترین سرویس کنترلی از نظر پروتکل سیپ می باشد، جهت برقراری مکالمات با برنامه های صوتی (سرور رسانه) و همچنین برقراری تماس های داخلی می باشد.
سرولت پیاده سازی این سرویس ProxyServlet می باشد که با استفاده از سرویس رجیسترار مقصد های مورد نظر برای پراکسی را پیدا کرده و پیام مورد نظر را به آن ها هدایت می کند. سرولت های کال منجر برای استفاده از این سرویس می بایست متد doProxy را فراخوانی کنند.
در مدل پراکسی، به ازای هر مکالمه یک لگ بیشتر درگیر نمی باشد که آن هم توسط سرور سیپ مدیریت می شود. از همین جهت، شی لگ در سرویس پراکسی تغییر وضعیت بسیار ساده ای دارد. نمودار حالت زیر، چگونگی تغییر وضعیت شی مکالمه را در برقراری ارتباط به روش پراکسی را نمایش می دهد. هر جایی که لگ مربوطه نیز تغییر وضعیت داشته، وضعیت جدید آن نیز (بعد از حرف "/") آورده شده است.

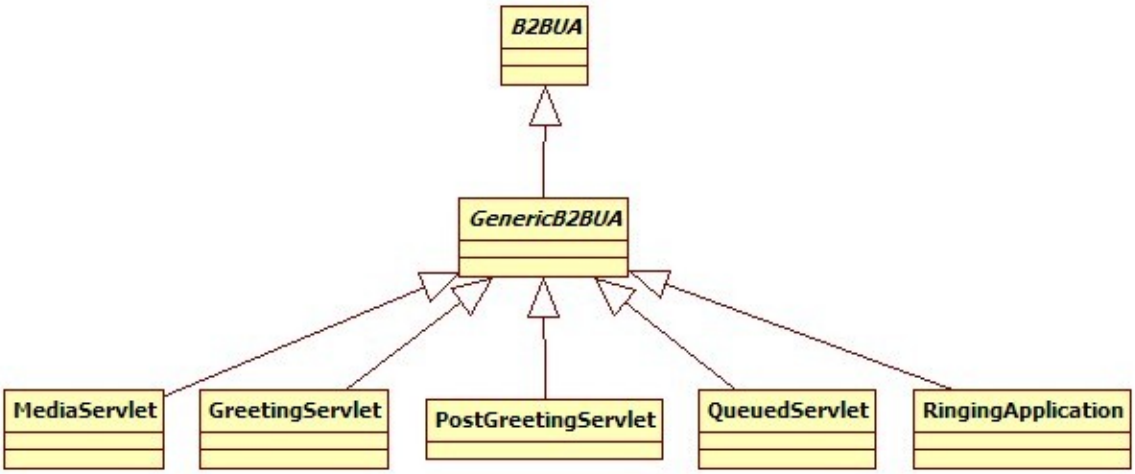


شکل ۱۴. نمودار حالت شی مکالمه در سرویس پراکسی

۶.۴.۲ سرویس صف

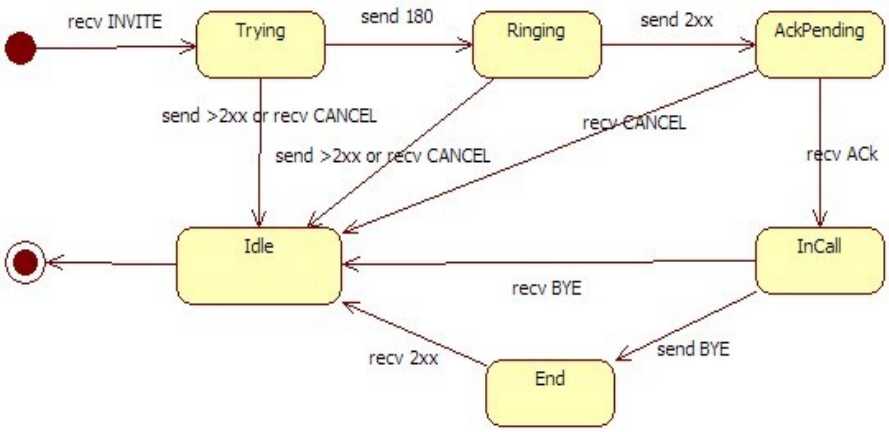
این سرویس جهت برقراری و کنترل تماس ها به اپراتورها می باشد. برای این منظور از ماشین های کنترلی مکالمات[23] استفاده می شود، امثال این ماشین ها برای پخش پیغام خوش آمدگویی و یا انتظار مکالمه بکار می روند. سرولت B2BUA، کلاس پایه ساختار ماشین ها می باشد که دو گروه از متد های مختلف را (یکی برای UAC و دیگری برای UAS) معرفی می کند که با پیشوند های doCallerxxx و doCalleexxx متمایز می شوند. به این ترتیب این متد ها بر اساس نوع پیغام رسیده به ماشین مورد نظر (رویداد ورودی) و اینکه طرف پیغام UAS باشد و یا UAC فراخوانی می شوند. پس آن چه این متد ها معمولاً انجام می دهند بررسی و تصمیم گیری بر اساس وضعیت فعلی (بر اساس شی مکالمه و طرفین درگیر) و انجام عملیات لازم و در نهایت تغییر وضعیت به وضعیت جدید می باشد. با این طراحی مطالعه جزییات از روی کد ماشین ها امکان پذیر بوده و می توان به سادگی با رجوع به متد های مربوطه جزییات تصمیم گیری در یک حالت خاص در یک ماشین خاص را پی گیری کرد. نمودار انواع سرولت های B2BUA را در زیر می بینید.

23 Back to Back User Agent (B۲BUA)

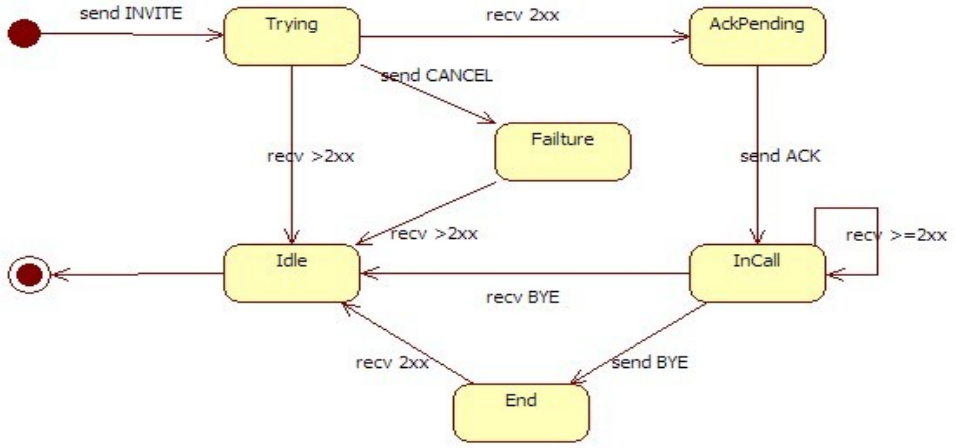


شکل ۱۵. نمودار کلاس های مربوط به سرولت های B2BUA

نمودارهای حالت زیر، چگونگی تغییر وضعیت **دو** طرف درگیر در یک ماشین ساده برای اتصال به اپراتور بدون پخش شدن پیام خوشامدگویی و یا انتظار را نمایش می دهد. ماشین های کال منجر در واقع گسترشی بر این دو نمودار حالت پایه هستند به این ترتیب که وضعیت ها و انتقال های جدیدی را به این ماشین افزوده و یا انتقال های موجود را بر اساس نیاز خود تغییر داده اند. با توجه به طراحی متد های مشخصی برای تصمیم گیری و به روز رسانی وضعیت لگ ها (در کلاس Leg)، و همچنین مدیریت تماس ها (در کلاس CallService) و مدل ماشین های B2BUA که در بالا آورده شد، مطالعه جزییات ماشین ها از روی کد در مواقع نیاز کار سختی نخواهد بود.



شکل ۱۶. نمودار حالت لگ UAS در برقراری یک تماس معمولی



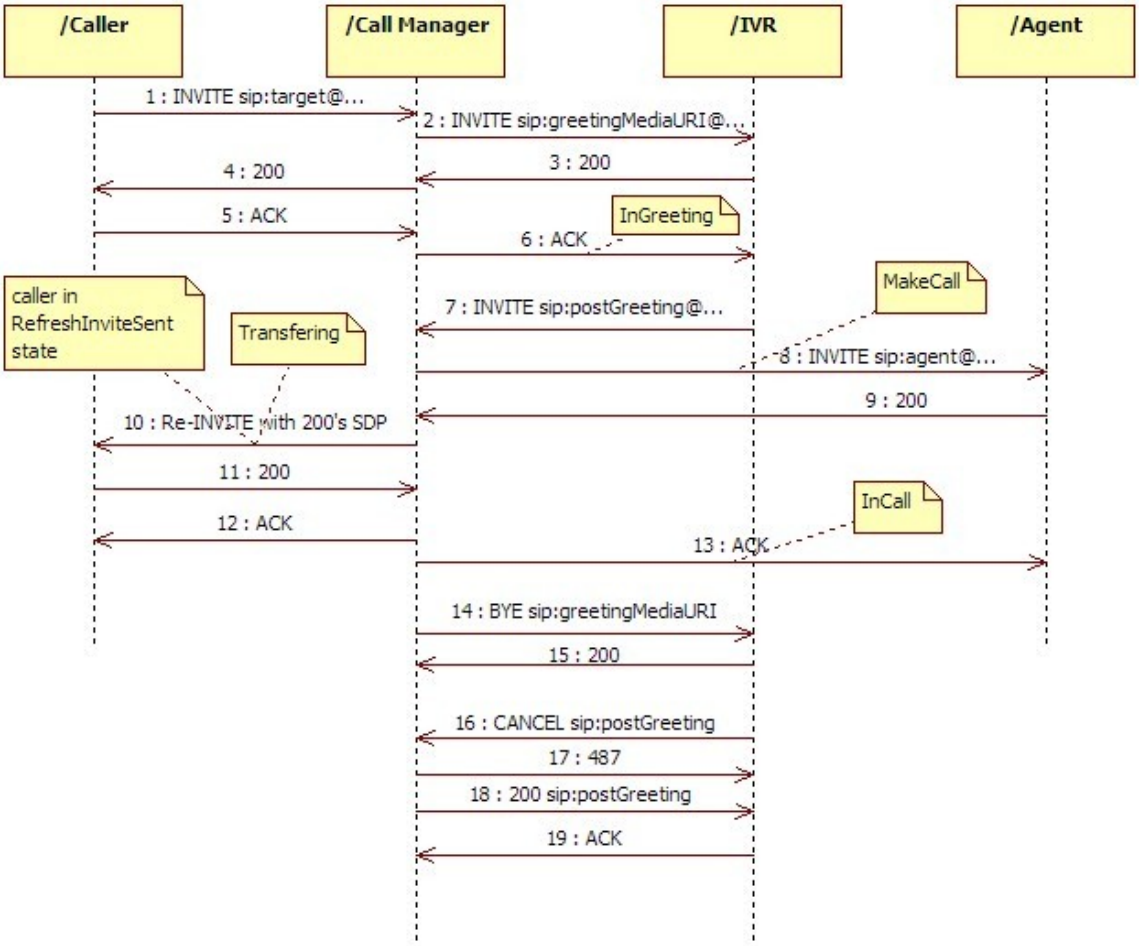
شکل ۱۷. نمودار حالت لگ UAC در برقراری یک تماس معمولی

۵. ۶.۴.۲.۱ سناریوی اتصال به اپراتور (با پخش پیام خوش آمدگویی)

مکالماتی که باید توسط اپراتور پاسخ داده شوند، قبل از اتصال مستقیم به اپراتور، جهت پخش پیام خوش آمد گویی اپراتوری (مثل راهنمای اپراتوری) ابتدا با سرور رسانه درگیر می شوند و سرور رسانه بعد از پخش سند صوتی مربوطه[24] که آن را از کال منجر دریافت می کند، اقدام به انتقال مکالمه به اپراتور مربوطه می کند. این از طریق یک پیام جدید INVITE از جانب سرور رسانه با نام کاربری "*postGreeting*" انجام می شود. کال منجر با دریافت این پیام (مورد ۳ قسمت ۶.۱.۳ را ببینید)، فراخوان مکالمه را به اپراتور متصل کرده و ارتباط با لگ های درگیر با سرور رسانه را خاتمه می دهد. در نمودار توالی پیام زیر می توانید این سناریو را به جزییات ببینید. برای کامل تر بودن و در عین حال مفهوم تر بودن جزییات، تغییر وضعیت شی مکالمه مربوطه در قالب یادداشت[25] های UML آمده است. نمودار وضعیت لگ های درگیر در این سناریو مانند نمودار های وضعیت آورده شده در بالا هستند. فقط لگ فراخوان، یک وضعیت جدید را در پیام شماره ۱۰ به خود می گیرد که در نمودار آورده شده است.

---

24 Voice XML
25 Note



شکل ۱۸. نمودار توالی پیام های سیپ در سناریوی اتصال به اپراتور

۶.۵ درگیری برنامه های صوتی با مکالمات

یکی از پرکاربردتر ین نیازهای نرم افزار های مدیریت و کنترل مکالمات، ارتباط با منو و برنامه های صوتی در خلال مکالمات می باشد. پخش پیام های خوش آمدگویی، کد اپراتوری، منوهای تعاملی صوتی و .... از این کاربردها می باشند. این روزها با رشد تکنولوژی های موجود در بستر VOIP، این امکان قابل انتظار است که مدیران مراکز ارتباطی بتوانند نحوه درگیری مشتریان را با برنامه های صوتی به طور پویا تعریف کنند. یعنی کال منجر باید این قابلیت را داشته باشد تا بسته به قوانین و مقررات و برنامه های داخلی هر مرکز ارتباطی و اپراتوری، به صورت معین و منظم این برنامه ها را دریافت و در کنترل مکالمات اعمال کند. به عنوان مثالی برای این کاربرد، فرض کنید مرکزی قصد پخش منوهای متفاوت برای مشتریان مناطق متفاوت شهری را دارد، یا یک روابط عمومی قصد پخش منوهای صوتی جایگزین برای ساعات خاصی از شبانه روز برای مشتریان خود دارد. البته نیازها و کاربردهای بسیار پیچیده تری را نیز می توان متصور شد.

در زمانی که نرم افزار کال منجر پی ریزی می شد، این نیازها بسته به موقعیت پروژه از نظر زمان و تجربه، دیده نشده بودند، بنابراین مدل سازی و پیاده سازی این روش به کمال، در حال حاضر زمان و تلاش خاصی را می طلبد، اما مدل پیشنهاد شده فعلی و چند نمونه الگویی که بر اساس همان مدل در کال منجر پیاده سازی شده است نیاز های کنونی را پاسخ می دهد. در ادامه این مدل را شرح می دهیم.

با توجه به امکانات کال منجر و سرویس پروفایل بهترین روش برای بیان قوانین درگیری برنامه های صوتی با مکالمات، پروفایل ها می باشند. برای این منظور شمای پروفایل ها را چنان گسترش دادیم تا امکان بیان قوانین برنامه های را نیز داشته باشد. برای این منظور تگ ApplicationIntegration را به پروفایل اپراتور ها و استخر های اپراتوری[26] اضافه کردیم. این تگ را با مثالی توضیح می دهیم.

```
<ApplicationIntegration>
   <Application name="Menu08" expression="exp" priority="9">
        <Participation party="caller" role="required"/>
        <Participation party="agent" role="teardown"/>
        <Parameter name="param01" value="value01"/>
        ...
   </Application>
   ...
</ApplicationIntegration>
```

در این تگ لیستی از برنامه های ممکن تک تک بوسیله تگ Application آورده می شود. نام هر برنامه صوتی، عبارت فعال سازی آن (به نوعی قانون فعال شدن برنامه) و اولویت برنامه در ابتدا به عنوان صفات این تگ می ایند. داخل بدنه این تگ نیز چگونگی درگیر شدن برنامه با مکالمه برای کال منجر بیان می شود و در انتها اگر نیاز باشد پارامتر هایی که باید به برنامه فرستاده شوند آورده می شوند.

به این ترتیب، خواهیم توانست برنامه های متنوعی را با قوانین فعال سازی توانمندی به طور پویا و در زمان اجرا به همراه مکانیزم کنترلی آن از دید طرف های درگیر در مکالمه برای کال منجر تعریف کنیم. حال به تفضیل قدرت پشتیبانی کال منجر را در بیان قوانین فعال سازی و سمنتیک کنترلی درگیری برنامه شرح می دهیم.

---

26 PoolTarget Profiles

۶.۵.۱ عبارت قانون فعال سازی

برای بیان قوانین درگیری برنامه ها از عبارت های منطقی استفاده می شود که مقدار نهایی آن ها درست یا غلط باشد. در صورت برقرار بودن عبارت منطقی، برنامه مورد نظر شرایط درگیر شدن را دارد و کال منجر مکالمه را با آن برنامه مطابق سمنتیک بیان شده در ادامه درگیر می کند. برای پیاده سازی این عبارات به یک زبان تعریف شده نیاز داریم که بتوانیم عبارات متنوع منطقی را با آن ها بیان کنیم. ما برای این منظور بهترین پاسخ را زبان عبارات[27] موجود در زبان جاوا دیدیم. این زبان امکانات قابل ملاحظه ای برای پاسخ گویی به نیاز های کال منجر دارد. از این قبیل اپراتور های منطقی و ریاضی، دسترسی به شی های تعریف شده در زمان اجرا، فراخوانی توابع آن ها و ... را می توان نام برد. بنابراین مقادیر صفت expression در واقع همین عبارات هستند که قواعد نوشتن آن ها را در زیر می بینیم.

از ابتدایی ترین امکانات این زبان اپراتور ه و ثابت های آن می باشد که لیست آن را در جدول زیر می بینید.

اپراتور (عمل گر) با ثابت	شرح
and یـا &&	اپراتور منطقی
or یـا ||	اپراتور عطف منطقی
not یـا !	اپراتور نقیض
==	اپراتور تساوی
=!	اپراتور نا مساوی
=>، =<، >، <	اپراتور های کوچکتر/بزرگتر (مساوی)
+، -، *، /، div، mod، - (منفی)	اپراتور های ریاضی
۰..۹	اعداد و ارقام، برای اعشار از نقطه استفاده می شود
"Hello"	عبارات حرفی
True و false	ثابت های منطقی درست و نادرست
null	ثابت null

جدول ۶. اپراتور های قابل استفاده در زبان عبارات قوانین فعال سازی

اما از مهم ترین و کاربردی ترین امکانات این زبان استفاده از شی های تعریف شده و دسترسی به صفات و متد های آن هاست. برای دسترسی به یک صفت از شی، از نقطه (مانند زبان های برنامه نویسی) استفاده می شود و

27 Expression Language

فراخوانی متد ها نیز شکلی شبیه به زبان های برنامه نویسی دارد. برای استفاده از این امکانات باید شی های تعریف شده در کال منجر را بدانیم تا بتوانیم قوانین بدرد بخوری بنویسیم. در جدول زیر این اشیا را توضیح داده ایم.

نام شی	شرح	مثال
call	شی مکالمه، به مکالمه کنونی اشاره می کند. می توان از تمام صفات شی CallService که متد های getter و setter دارند استفاده کرد.	call.arrival, ...
request	پیام INVITE اولیه فراخوان (از نوع کلاس SipServletRequest)	request.requestURI request.to, request.from, ...
this	شی هدف مکالمه (کلاس CallTarget) به شی ای پروفایل اشاره می کند.	this.telephoneNumber this.name this.queueDepth
date	شی زمان کنونی سیستم (از جنس java.util.Date)	date.hours, ... date.year, date.month, ...

جدول ۷. شی های تعریف شده برای استفاده در عبارات قوانین فعال سازی

۶.۵.۱.۱ شی تایمر

برای استفاده از تایمر در عبارات فعال سازی برنامه ها، شی مجزایی به نام timer در نظر گرفته شده است که با استفاده از متد create آن می توان تایمری را بر حسب میلی ثانیه تولید کرد، اما برای اینکه حاصل این عبارت مانند بقیه عبارات قانون فعال سازی می بایست یک عبارت منطقی باشد، از صفت منطقی timeout استفاده می شود که مقدار این صفت وقتی زمان تایمر هنوز سر نرسیده نادرست و بعد از آن زمان درست می باشد. برای روشن شدن مطلب به مثال زیر توجه کنید.

```
timer.create(10000).timeout
```

این شرط زمانی برقرار می شود که تایمر ده ثانیه ای مربوطه سر رسد.

۶.۵.۲ سمنتیک کنترلی مکالمه

برای اینکه بتوانیم سمنتیک کنترل مکالمه را برای کال منجر بیان کنیم تا نرم افزار بتواند مکالمه را به شکل صحیح با برنامه صوتی در گیر کند تگ Participation را تعریف کردیم که بیان گر نحوه حضور یا برخورد با طرف های درگیر در مکالمه را نشان می دهد. برای این کار دو صفت یکی بیانگر شناسه طرف مورد نظر (برای

آدرس دهی طرف های درگیر در مکالمه) و دیگری بیانگر وضعیت طرف در درگیری با برنامه، تعریف کرده ایم. البته این تعریف ها نا بالغ به نظر می رسد و شاید به برخی کاربرد های خاص محدود شود اما فعلاً پاسخگوی نیاز هاست. در ادامه مقادیری که هر یک از این دو صفت می توانند داشته باشند را شرح می دهیم. این مقادیر در واقع نام نقش طرف های درگیر در مکالمات هستند که در شرایط مختلف در یک مکالمه حاضر می شوند. این نام ها در کلاس Leg تعریف شده اند و بسته به نیاز ها می توانند گسترش نیز پیدا کنند.

مقدار صفت party	شرح
Caller	طرف فراخوان مکالمه
Agent	طرف اپراتور
Waiting Media	طرف ملودی انتظار
Greeting Media	طرف پخش پیام خوش امد گویی اپراتور
IVR Transferer to Agent	طرف سرور رسانه ای در اتصال مکالمه به اپراتور
Transferee to Greeting	طرفی که بعد از ترنسفر یک مکالمه به اپراتور برای پخش پیام خوش امد گویی به سمت سرور رسانه ایجاد می شود
Voice Application	طرف برنامه صوتی در اتصال به سرور رسانه

جدول ۸. معرفی مقادیر فعلی که صفت party می تواند به خود بگیرد

مقدار صفت role	شرح
Required	طرف مورد نظر باید در اتصال به برنامه صوتی حاضر باشد (با مکانیزم b۲bua به صورت مستقیم متصل شود)
Teardown	ارتباط طرف مورد نظر باید از مکالمه قطع شود

جدول۹. معرفی مقادیر فعلی که صفت role می تواند به خود بگیرد

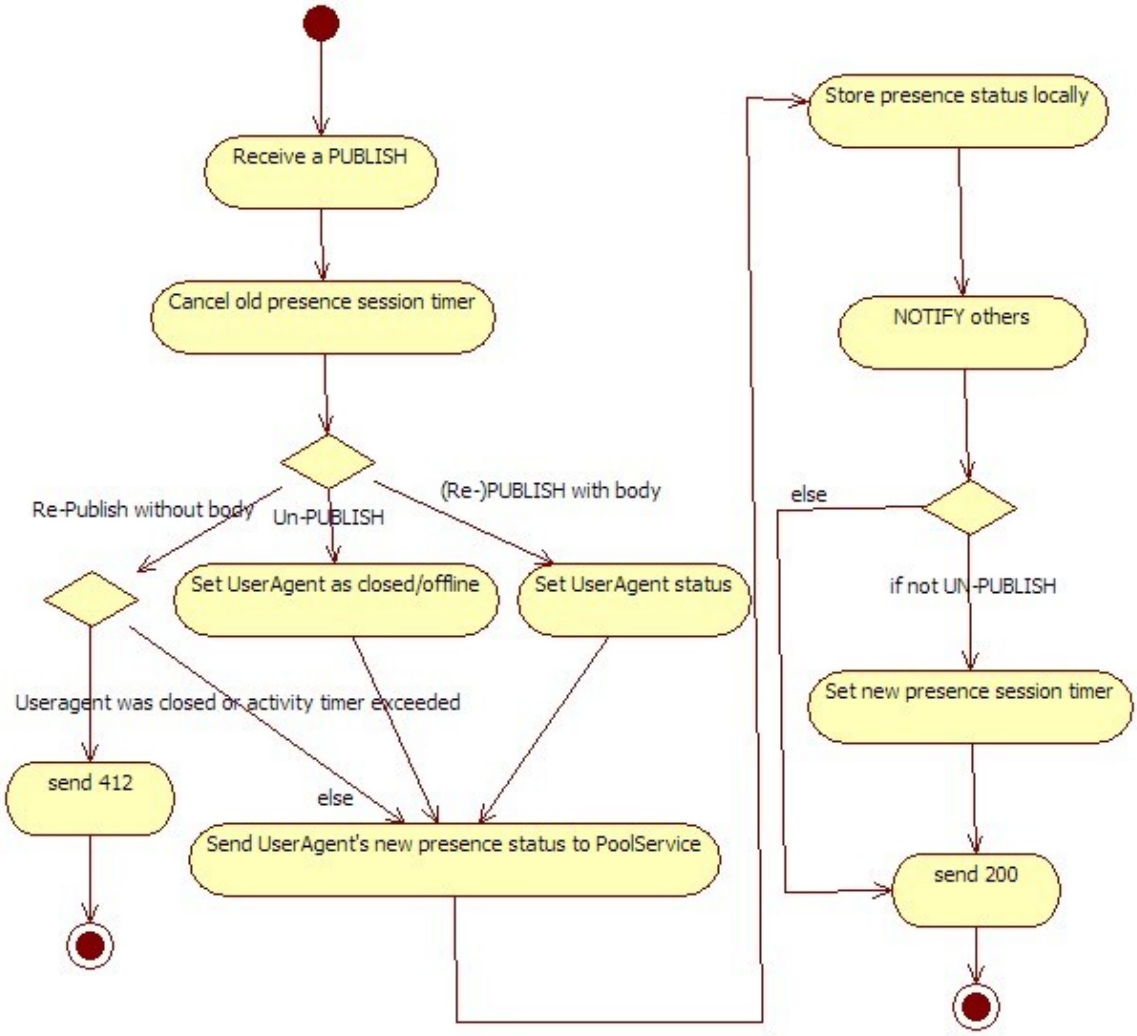
۶.۵.۳ نحوه استفاده از سرویس

برای این منظور در کلاس CallTarget متدی به نام assignApplications قرار داده شده است، که با بررسی پروفایل مربوطه، امکان درگیری برنامه های موجود در پروفایل را تک تک با استفاده از قوانین فعال سازی آن ها چک می کند و در صورت اول تطابق، برنامه صوتی مربوط را درگیر مکالمه می کند. اما کدام مکالمه؟ این روند در چه زمانی اجرا می شود؟ در حال جاضر دو مدخل در کال منجر تعریف شده ا ند که روند گفته شده برای آن ها اعمال می شود. اولین مدخل، مرحله ایست که صف هدف یک مکالمه تشخیص داده شده و قبل از وارد شدن مکالمه به صف! و دومین مدخل درست زمانی است که یک مکالمه توسط کال منجر به اپراتوری منتسب می شود (یعنی حتی قبل از اتصال مکالمه به اپراتور).

۶.۶ سرویس اعلان وضعیت[28]

شامل سرولت های سیپ برای دریافت و سرویس دهی به پیغام ه ای NOTIFY، PUBLISH و SUBSCRIBE جهت اعلام حضور و اعلان وضعیت اپراتورها و انواع موجودیت های موجود طرف تماس می باشد. سرولت مربوط به این سرویس PresenceServlet و سرویس متناظر آن PresenceService می باشند. این سرولت پیام های PUBLISH را به عنوان رویدادی برای تعیین وضعیت اپراتورها، دریافت می کند و مطابق نمودار فعالیت زیر عمل می کند. پیام های SUBSCRIBE نیز برای ایجاد لیست های وضعیت حضور اپراتوری استفاده می شوند تا مدیران بتوانند از وضعیت اپراتور های فعال در سیستم در هر لحظه با خبر باشند. انجام این خبررسانی توسط پیام های NOTIFY، به دو شکل، صورت می گیرد. چه بعد از پیام PUBLISH رسیده از یک موجودیت که پیام NOTIFY توسط کال منجر به تمامی افرادی که برای او خود را ثبت نام[29] کرده اند و چه در زمانی که کسی پیام SUBSCRIBE برای کال منجر می فرستد، در پاسخ پیام NOTIFY شامل وضعیت کنونی موجودیت مقصد ثبت نام را از کال منجر دریافت می کند.

28 Presence Service
29 Subscribe



شکل ۱۹. نمودار فعالیت سرویس اعلان وضعیت هنگام دریافت پیام PUBLISH

۶.۶.۱ چرخه حیات اپراتور

هر شی از کلاس Agent، نماینده حضور یک اپراتور در سیستم است. بنابراین به ازای ورود هر اپراتور یک شی Agent توسط کلاس AgentService ایجاد شده، اطلاعات مورد نیاز آن از روی سرور دایرکتوری خوانده می شود و شی Agent مذکور به استخرهای عضویت اش، اضافه می شود. این شی تا زمانی که اپراتور از سیستم خارج شود (Logoff کند) در حافظه وجود خواهد داشت و دائماً توسط رویدادهای QMS و یا Presence در حال به روز شدن است.

۶.۷ سرویس رجیستر[30]

سرولت ورودی برای دریافت پیغام های REGISTER جهت سرویس رجیستری کال منجر را شامل می شود. سرویس رجیسترار برای یافتن آدرس فیزیکی اپراتورها و سرور رسانه ای از روی آدرس رکورد آن ها به کار می رود.

۷. مؤلفه گزارشات (Logging)

شامل کلاس های لازم برای سرویس گزارش دهی کال منجر است. این سرویس همانطور که گفته شد تیم LogService نام دارد. گزارشات فعلی در قالب فایل های XML روی سرور WEBDAV نوشته می شوند که شمای آن ها توسط فایل opxiActivityLog.xsd تعریف شده است. از روی شمای مذکور، کلاس های جاوایی بصورت اتوماتیک ساخته می شوند که کال منجر از آن ها برای ایجاد و دسترسی به ساختار گزارشات خود استفاده می کند. این گزارشات در هنگام ثبت شدن روی سرور Exchange توسط پیاده سازی هایی از کلاس LogReportDAO به قالب XML در آمده و توسط پروتکل Webdav روی سرور نوشته می شوند. در حال حاضر دو کلاس به نام های ExchangeLogReportDAO و ExchangeTempLogReportDAO که اولی برای نوشتن گزارشات نهایی و دومی برای ثبت دوره ای گزارشات موقتی استفاده می شوند، کلاس LogReportDAO را پیاده سازی می کنند.

گزارشات کال منجر در حال حاضر از دو نوع می باشند، گزارش کارکرد اپراتورها و گزارش کارکرد سیستم که شمای آن ها در فایل گفته شده تعریف شده است. علاوه بر کلاس های اشیا گزارشات که به طور خودکار از روی شما ایجاد می شوند، کلاس هایی برای مدیریت این اشیا و در واقع انجام منطق گزارش گیری در نظر گرفته شده است. کلاس پدر هر واحد (یا نوع) گزارش OpxiActivityLogger می باشد. هر نوع گزارش با ارث بری از این کلاس و پیاده سازی منطق مطلوب قابل افزوده شدن است. اما همانطور که در شمای گزارش کارکرد سیستم هم می بینیم، برخی گزارشات در دل خود قلم گزارش های دیگری را جای داده اند، مثل گزارش برنامه های صوتی. این نوع گزارشات از نوع ChildActivityLogger در نظر گرفته می شوند و برای افزودن قلم های گزارش جدید در ساختار گزارش کارکرد سیستم بکار می روند. مدیریت و منطق گزارش گیری این گزارشات از طریق افزودن متد های جدید در کلاس گزارش گیری اصلی (از نوع OpxiActivityLogger) انجام می شود.

برای افزودن واحد گزارش گیری جدید، بعد از تعریف کلاس جدیدی از نوع OpxiActivityLogger، لازم است متد های چرخه حیات گزارش مربوطه را پیاده سازی نمایید. این متدها به شرح زیر می باشند.

```
protected abstract OpxiActivityLog initLogVO( final Serializable object );
```

این متد برای ایجاد اشیای اولیه گزارشات و مقدار دهی اولیه آن ها می باشد. برای مثال به کد مراجعه کنید.

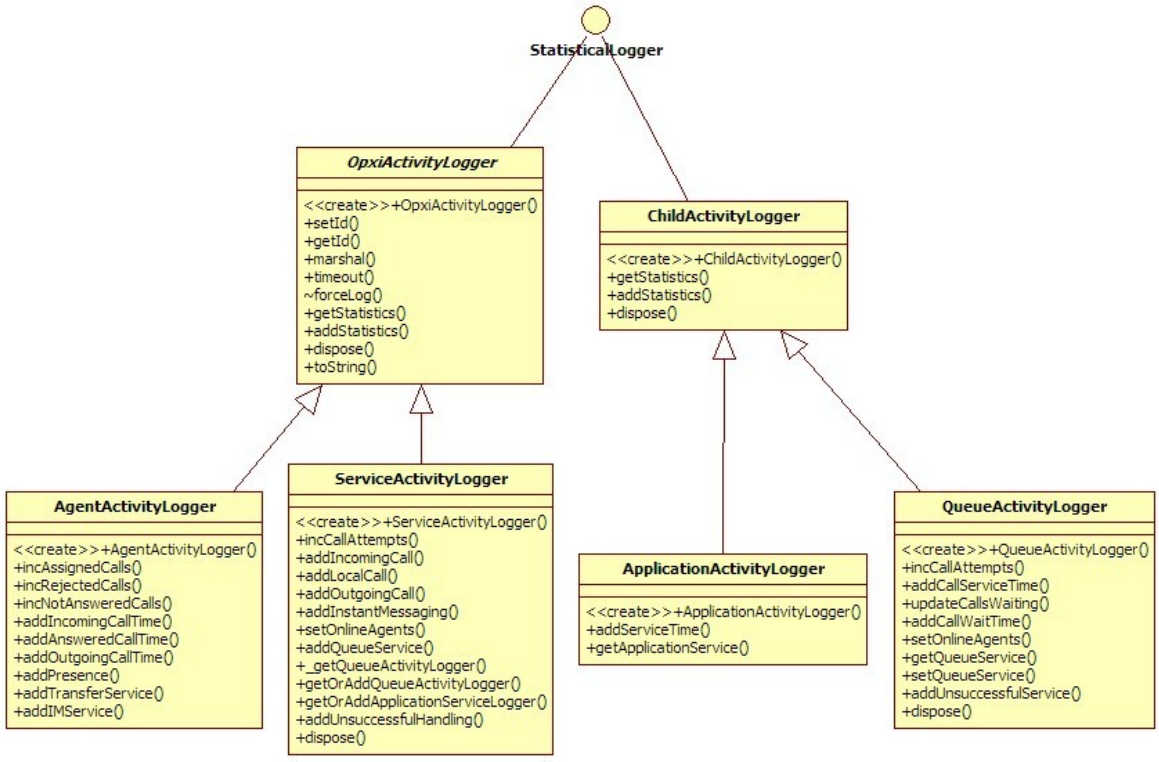
```
protected abstract void doForceSpecific() throws ForceActionException;
```

این متد عملیات ثبت نهایی گزارش مربوطه را پیاده سازی می کند.

30 Location Service

```
protected abstract void beforeForceLog( Object cause );
```

و بالاخره این متد قبل از انجام ثبت نهایی گزارش فراخوانی می شود تا کنترل بیشتری به واحد گزارش گیری برای بستن و نهایی سازی گزارشات بدهد. درهنگام فراخوانی این متد گزارشات هنوز در حافظه قرار دارند.
در زیر نمودار کلاس مولفه گزارشات را می بینید.



شکل ۲۰. نمودار کلاس انواع لاگرها

۸. مؤلفه وب

شامل فایل های JSP، سرولت های وب و وب سرویس های نرم افزار است. بخش وب، در حال حاضر بسیار ساده بوده و با تعامل مستقیم سرولت ها و JSPها پیاده سازی شده است. برای گسترش لایه وب کال منجر، چارچوب Struts انتخاب شده است و به همین منظور فایل های جار و پیکربندی این چارچوب به کال منجر افزوده شده است ولی به علت نهایی نشدن استفاده از این چارچوب، در اینجا به شرح آن نمی پردازیم. در این قسمت سرولت های وب کنونی فعال در کال منجر را معرفی می کنیم.

۸.۱ سرولت های پایه

۸.۱.۱ سرولت OpxiHttpServlet

این کلاس نقشی شبیه به همتای سیپ خود (OpxiSipServlet) دارد. این سرولت دو متد اساسی برای دسترسی به SipFactory و ServiceFactory را تعریف می کند. بقیه سیپ سرولت ها برای استفاده از سرویس ها باید از این سرولت ارث بری کنند.

۸.۱.۲ سرولت های تولید Voice XMLها

دو سرولت GreetingVxmlServlet و WaitingRoomVxmlServlet که در پکیج vxml قرار دارند برای تولید فایل های vxml لازم برای سرور رسانه ای جهت پخش پیام خوش امدگویی اپراتور و ملودی انتظار به کار می روند.

۸.۱.۳ سرولت WebIMServlet

این سرولت برای فرستادن پیام متنی فوری از طریق کنسول وب مورد استفاده قرار می گیرد.

۸.۱.۴ سرولت WebTransfer

این سرولت نیز برای انتقال مکالمات فعال از طریق کنسول وب مورد استفاده قرار می گیرد.

۸.۱.۵ سرولت A2CBridge

این سرولت برای هدایت اپراتور به سمت صفحه خانگی فراخوان مکالمه که بر روی نرم افزار مدیریت ارتباط با مشتری[31] قرار دارد، نوشته شده است.

۸.۲ وب سرویس ها

یکی از واسط های کنترلی کال منجر، مخصوصا برای مدیریت پروفایل ها از طریق نرم افزارOpxiManager، وب سرویس ها هستند. پیاده سازی وب سرویس ها در کال منجر مبتنی بر استاندارد J2EE ۱.۴ صورت گرفته تا این وب سرویس ها بر روی هر سروری که از این استاندارد پشتیبانی می کند بدون تغییر قابل اجرا باشند.

31 Customer Relationship Management (CRM)

۸.۲.۱ ساختار وب سرویس ها

کلاس های پیاده سازی وب سرویس باید از واسط javax.xml.rpc.server.ServiceLifecycle ارث بری کنند و متد های init و destroy را مطابق نیاز پیاده سازی کنند. کلاس AdminServiceImpl در کال منجر همین کار را انجام داده است. این کلاس به عنوان پیاده سازی تنها وب سرویس کال منجر می باشد. اما سرور برای بارگزاری و اجرای این وب سرویس نیاز به اطلاعات بیشتری در مورد آن دارد. این اطلاعات (بر اساس استاندارد وب سرویس در J2EE ۱.۴) شامل فایل های jaxrpc-mapping، wsdl و webservices.xml می باشد که هر سه توسط تسک های Ant موجود و به کمک چارچوب XDoclet به طور اتوماتیک تولید می شوند. این تسک ها را می توانید در بخش ۸ ببینید. اما برای انجام این منظور ما باید اطلاعات مورد نیاز وب سرویس برای ایجاد این فایل ها را داخل کد کلاس وب سرویس (این کلاس حتما باید در پکیج com.basamadco.opxi.callmanager.web.services ایجاد شود) و به صورت تگ های XDoclet و در قالب Javadoc، فراهم کنیم. این تگ ها در دو سطح تعریف می شوند. یکی در سطح کلاس پیاده سازی وب سرویس و برای تعریف مشخصات اصلی وب سرویس و دیگری در سطح متد و برای تعریف متد های وب سرویس.

به نمونه کد زیر توجه کنید.

```
/**
 * @web.servlet
 *              name="AdminService"
 *              service-endpoint-
class="com.basamadco.opxi.callmanager.services.AdminService"
 *              load-on-startup="1"
 *
 * @web.servlet-mapping
 *              url-pattern="/services/AdminService"
 *
 * @wsee.port-component
 *                      name="AdminService"
 */
```

در تکه کد بالا، چهار مشخصه برای این وب سرویس تعریف شده اند که مفهوم آن ها به ترتیب زیر می باشد.

نام مشخصه	مفهوم مشخصه
name (در قسمت @web.servlet)	نام وب سرویس
service-endpoint-class	نام کلاسی که به عنوان واسط سرویس[32] به صورت اتوماتیک تولید خواهد شد.
url-pattern	آدرس نسبی دسترسی به وب سرویس روی سرور
name (در قسمت @wsee.port-component)	نام پورت وب سرویس (همنام با نام وب سرویس)

جدول ۱۰. مفهوم مشخصه های تگ های معرفی وب سرویس

همانطور که از روی کد قابل تشخیص است، نوع سرولت را برای پیاده سازی وب سرویس های کال منجر توسط سرور J۲EE انتخاب کرده ایم. (گزینه دیگر استفاده از EJB برای پیاده سازی وب سرویس بود.)

اما برای معرفی متد های وب جدید و قابل فراخوانی در این کلاس از نمونه کد زیر برای آن متد استفاده می کنیم.

```
/**
 * @web.interface-method
 *
 * @ejb.interface-method
 */
```

این دو تگ (هر دو) برای ایجاد صحیح فایل های گفته شده جهت اجرای وب سرویس لازم اند، هر چند که وب سرویس بصورت EJB پیاده سازی نشده است. این نکته به تجربه و با بررسی جزییات تسک های Ant مربوطه بدست آمده است که ناشی از عدم بلوغ ابزار لازم برای ایجاد اتوماتیک استاندارد وب سرویس ها در محیط جاوا می باشد.

۹. ملاحظات پیاده سازی

تا کنون با طراحی سطح بالای کال منجر، مولفه های آن و سرویس هایشان و چگونگی تعامل بین آن ها آشنا شده ایم. در این بخش به توضیح برخی جنبه های سطح پایین تر و مربوط به پیاده سازی، مانند چگونگی پیاده سازی انحصار دوجانبه برای کنترل همروندی خواهیم پرداخت.

32 Service Endpoint Interface (SEI)

۹.۱ مکانیزم برقراری انحصار دوجانبه

برای کنترل همروندی در کال منجر از مکانیزم قفل گذاری استفاده شده است. برای این منظور از امکانات خود زبان جاوا[33] استفاده شده است. شی هایی که به عنوان قفل عمل می کنند نیز توسط کلاس LockManager تولید می شوند. متد های این کلاس موظف هستند برای ورودی های یکسان، همواره یک شی قفل یکتا را در اختیار فراخوان برای استفاده در بلوک synchronized قرار دهند، تا برقراری انحصار دو جانبه بین بخش های کد تضمین شود. در کال منجر به طور کلی از دو سری قفل استفاده می شود:

۱. **قفل های محلی کلاس ها:** این ها مواردی هستند که شی قفل داخل کلاسی که مورد نیاز است تعریف شده است و برای دسترسی به آن، از کلاس LockManager استفاده نمی شود. قفل هایی که برای دسترسی های محلی به ساختمان داده های داخلی کلاس ها، مثل لیست ها، صف ها و ... استفاده می شوند، از همین نوع هستند. برای پیاده سازی این قفل ها معمولاً از یک شی ثابت استفاده می شود. به مثال زیر از کلاس Queue توجه کنید:

```
private final Object CALL_SCHEDULE_LOCK = new Object();
```

این قفل برای کنترل همروندی در عمل خروج مکالمات از صف برای اتصال به اپراتور[34] استفاده می شود.

```
public CallService scheduleCallFor( String agentAOR ) ... {
    synchronized( CALL_SCHEDULE_LOCK ) {
        ...
    }
}
```

به این ترتیب در هر واحد زمانی یک و فقط یک پروسه در حال اجرای این عمل می باشد.

۲. **قفل های سراسری:** این ها قفل هایی هستند که توسط کلاس LockManager تولید و مدیریت می شوند و برای برقراری انحصار دوجانبه بین کد های کلاس های مختلف استفاده می شوند. در اینجا قفل های موجود در کال منجر از این نوع را با بیان کاربرد آن ها لیست کرده ایم.

۱.۲. **قفل سشن:** این قفل توسط متد LockManager.getSessionLock بر روی شی سیپ سشن تولید می شود و برای برقراری انحصار دو جانبه بین فراخوانی متد های doRequest و

33 Synchronized Blocks
34 Call Scheduling

doResponse در کلاس OpxiSipServlet استفاده می شود. به این ترتیب پیام هایی که در یک سشن خاص به مولفه سیپ کال منجر می رسند، به طور متوالی اجرا و پاسخ داده می شوند.

۲.۲. قفل INVITE/CANCEL:

این قفل توسط متد LockManager.getInviteCancelLock بر روی شی سیپ سشن تولید می شود و برای جلوگیری از ایجاد مسابقه[35] در دریافت پیام های INVITE و CANCEL در یک سشن استفاده می شود. این قفل داخل متد doRequest کلاس OpxiSipServlet به کار گرفته شده است.

۳.۲. قفل CANCEL/۲۰۰:

این قفل توسط متد LockManager.getCancel_200_RaceLock بر روی شی مکالمه تولید می شود و برای جلوگیری از ایجاد مسابقه در دریافت پیام CANCEL فراخوان و پیام پاسخ ۲۰۰ فراخوانده شده، استفاده می شود. این قفل در کلاس B۲BUA و بین متد های doCallee200ToInvite و doCallerCancel انحصار متقابل برقرار کرده است.

۴.۲. قفل انتخاب اپراتور: این قفل توسط متد LockManager.getLockById و بر اساس آدرس اپراتور[36] تولید می شود و بین اعمال انتخاب اپراتور[37] (متد hunt در SkillBasedPool) و انتساب مکالمه بعد از آزادی اپراتور[38](متد serviceAgentIdleEvent در کلاس QueueManagementService) انحصار متقابل ایجاد می کند. اگر چنین نباشد، یک اپراتور ممکن است از طرف دو پروسه برای پاسخ دهی به مکالمات مختلف در آن واحد انتخاب شود. در واقع اپراتور ها برای پاسخ دهی به مکالمات منابع مشترک[39] هستند.

۵.۲. قفل Reject/Schedule:

این قفل توسط متد LockManager.getRejectScheduleLock و بر روی شی مکالمه تولید می شود. یک مکالمه در صف به انتظار به سر می برد تا زمانی که یا اپراتوری آزاد شود که قادر به پاسخ گویی به آن مکالمه باشد و یا زمان حد انتظار آن فرا رسد. این قفل جلوی همزمانی این دو اتفاق را می گیرد.

۶.۲. قفل Release/Schedule:

این قفل توسط متد LockManager.getReleaseScheduleLock و بر روی شی مکالمه ایجاد می شود و منطق آن تقریباً شبیه قفل بالاست با این تفاوت که این بار انحصار متقابل بین اعمال انتخاب مکالمه و آزاد شدن مکالمه صورت می گیرد. این انحصار متقابل بین

35 Race
36 Address Of Record
37 Agent Hunting
38 Agent Idle Time Call Scheduling
39 Shared Resource

متد های release از کلاس CallService و scheduleCallFor از کلاس Queue بر قرار شده است.

۱۰. مراحل کامپایل و ساخت فایل اجرایی

مراحل کامپایل و ساخت فایل قابل اجرای پروژه توسط نرم افزار Ant انجام می شود. برای این منظور یک فایل build.xml نوشته شده که وظایف مربوط به کامپایل فایل های جاوا ، ساخت اتوماتیک فایل های web.xml، آرشیوهای ear، sar و ... بقیه مستندات لازم را شامل می شود. مهمترین این وظایف موارد زیر هستند:

۱. build-ear: فایل قابل اجرای opxiCallManager.ear را داخل مسیر build می سازد .برای سخت عادی پروژه از این تسک استفاده کنید. این تسک بیشترین کاربرد را دارد.

توجه: جار فایل های متفرقه مورد نیاز در زمان اجرا، در شاخه lib/ext قرار دارند که در هنگام ساخت فایل اجرایی opxiCallManager.ear به داخل آن کپی می شوند تا در محیط اجرا مورد استفاده قرار بگیرند. برای این منظور هر فایل جار جدید که در زمان اجرا مورد نیاز است را در شاخه مذکور در پروژه کپی کنید.

۲. build: این تسک خود شامل مراحل موجود در build-ear می باشد . یعنی فایل opxiCallManager.ear را می سازد به علاوه اینکه فایل های پیکربندی مربوط به موجودیت های پایگاه داده در hibernate (فایل های hbm) را نیز ایجاد می کند. توجه داریم که در صورتی که فایل های هنوز hbm ایجاد نشده باشند (برای اولین بار) لازم است این تسک اجرا شود. یعنی اجرای حداقل یک بار این تسک اجباریست.

۳. build-all: این تسک علاوه بر دو مورد گفته شده ، مستندات مربوط به javadoc را نیز از روی فایل های جاوا می سازد و فایل اسکریپت ایجاد شمای پایگاه داده را نیز تولید می کند.

۴. clean: شاخه build و محتویاتش را پاک می کند. این شاخه همواره مجدداً قابل ایجاد است.

۵. compile-schemas: این تسک ، بعد از ایجاد فایل های جاوای مربوط به شمای XSD مربوط به موجودیت های پروفایل ها و گزارشات (از طریق تسک castor-src-gen)، آنها را کامپایل می کند و به صورت یک فایل Jar در مسیر Lib/ext کپی می کند.

۶. wscompile : این تسک کلاس واسط وب سرویس (SEI) و فایل های wsdl و jaxrpc-mapping را تولید می کند. شما به طور مستقیم از این تسک استفاده نمی کنید.

۷. wseedoclet: این تسک بعد از اجرای تسک قبل و برای تولید فایل webservices.xml بکار می رود.

۸. hbm-gen: برای تولید فایل های پیکربندی Hibernate (فایل های hbm) از روی کلاس های موجودیت های پایگاه داده استفاده می شود.

۹. sql-gen: برای تولید فایل اسکریپت شمای پایگاه داده بکار می رود.

۱۰. doc-gen: این تسک مستندات Javadoc را از روی کد تولید می کند.

۱۱ مراجع

[۱] J. Rosenberg, et. la., SIP: Session Initiation Protocol, RFC ۳۲۶۱, June ۲۰۰۲.

[۲] Anders Kristensen, SIP Servlet API Specification version ۱.۰, ۲۰۰۳.

[۳] Apache Ant ۱.۷.۰ Manual, available at http://ant.apache.org/manual

[۴] XDoclet online documentation available at http://xdoclet.sourceforge.net/xdoclet/index.html

[۵] HIBERNATE, Relational Persistence for Idiomatic Java, available at http://www.hibernate.org/hib_docs/v۳/reference/en/html

[۶] DAO Pattern, documentation available at http://java.sun.com/blueprints/corej۲eepatterns/Patterns/DataAccessObject.html

[۷] WebSphere® Application Server, Version ۶, Developing and deploying applications, December ۶, ۲۰۰۴

[۸] Web Services Handbook for WebSphere Application Server Version ۶.۱.

[۹] WebSphere Application Server V۶.۱, Technical Overview.