به نام خدا

معرفی شبیه ساز OPNET

۱- مقدمه

امروزه با رشد چشمگیر شبکه های کامپیوتری روشهایی که بتوان توسط آنها رفتار شبکه ها را آنالیز و بررسی کرد از اهمیت فراوانی برخوردارند. همچنین طراحان شبکه برای آزمایش ایده های نوین خود نیازمند ابزارهایی جهت شبیه سازی رفتار شبکه ها هستند. نرم افزارهای شبیه ساز ابزار مناسبی برای شبیه سازی شبکه ها هستند. بخصوص برای شبکه های امروزی که دارای ساختار با پیچیدگی بالا بوده و استفاده از روشهای آنالیز ریاضی در بررسی رفتار آنها عملا به صورت دقیق امکان پذیر نیست. از جمله مهمترین شبیه سازهای دانشگاهی[۱] شبکه می توان به موارد زیر اشاره کرد:

REAL, INSANSE, NetSim, Maisie, Ns-۲, U-Net

همچنین نرم افزارهای شبیه سازی تجاری متعددی نیز وجود دارند که از جمله مهمترین آنها می توان به شبیه سازهای **BONeS,COMNET ,OPNET** اشاره کرد. در اینجا به طور مختصر به هر یک از این شبیه سازها و کاربرد آنها اشاره می کنیم:

شبیه ساز **REAL** برای بررسی رفتار پویا و متغیر روشهای مختلف کنترل ازدحام و کنترل جریان در شبکه های داده با سوئیچینگ بسته ای به کار می رود. در این نرم افزار توپولوژی شبکه، پروتکل های مختلف و همچنین پارامترهای کنترلی به صورت یک سناریو تعریف می شوند. همچنین بیش از ۳۰ ماژول مختلف در این نرم افزار آماده شده است که می توان با استفاده از آنها رفتار پروتکلهای مختلف کنترل جریان و ازدحام را به صورت دقیق شبیه سازی و تقلید کرد.

شبیه ساز **INSANCE** برای بررسی رفتار الگوریتمهای مختلف در زمینه **IP** روی **ATM** طراحی شده است. شبیه ساز **NetSim** به منظور مدل کردن دقیق استاندارد اترنت و بررسی معضلات موجود در آن به کار می رود. شبیه ساز **Maisie** مبتنی بر زبان برنامه نویسی C بوده و برای شبیه سازیهای سلسله مراتبی مناسب است[۱].

شبیه سازهای شبکه را از دیدگاههای مختلف می توان تقسیم بندی کرد. یکی از این دیدگاهها برای تقسیم بندی شبیه سازهای شبکه روشی است که برای شبیه سازی به کار می برند. به طوریکه در حالت کلی شبیه سازی به دو روش متفاوت صورت می گیرد. روش اول شبیه سازی به کمک آنالیز دقیق رویدادها و روش دوم شبیه سازی رویدادهای گسسته[۲] است. در روش اول برای شبیه سازی از مدلهای ریاضی برای حصول نتایج استفاده می شود و شبکه با مجموعه ای از معادلات ریاضی مدل می شود. عیب عمده این روش شبیه سازی در ساده کردن بیش از حد شبکه در این مدلها و عدم توانایی در شبیه

---

۱ - Academic
۲ - Discrete event

سازی رفتار پویای شبکه ها است. اما در روش دوم شبیه سازی در سطح پایین (در سطح بسته های موجود در شبکه) انجام شده و از این رو نتایج از دقت بالایی برخوردارند. در این روش نتایج با شبیه سازی رویدادهای گسسته به دست می آیند. اما بر خلاف روش اول که شبیه سازی با سرعت بالایی انجام می شود، در این شبیه سازها نتایج با سرعت کمی حاصل خواهند شد[۲]. اما راه حل مناسب، استفاده از ترکیب این دو روش در شبیه سازی با هدف داشتن عملکرد مورد قبول و حفظ دقت مورد نیاز در موارد بحرانی است. این شبیه سازها را اصطلاحا شبیه سازهای مرکب یا دو رگه[۱] گویند. از مهمترین این شبیه سازها می توان به **OPNET**[۲] و **NS-۲**[۳] اشاره کرد. شبیه ساز **NS-۲** نسخه دوم شبیه ساز شبکه است که توسط **VINT**[۴] گسترش داده شده است. **NS-۲** یک شبیه ساز شبکه مبتنی بر رویداد است که به وفور توسط مهندسین شبکه مورد استفاده قرار می گیرد. در این شبیه ساز مدلهای مختلفی از اغلب پروتکلهای اینترنت را در خود دارد و همچنین به دلیل باز بودن منبع آن[۵] دارای سطوح مختلف پیکربندی بوده و همچنین قابلیت ایجاد پروتکلها و کاربردهای متداول را دارا می باشد.

شبیه ساز **OPNET** که در دانشگاه **MIT** و در سال ۱۹۸۷ ارائه شد، این امکان را برای کاربر فراهم می کند تا به طراحی و مطالعه شبکه های مخابراتی مختلف، ادوات شبکه، و پروتکلهای موجود در شبکه بپردازد. یکی از قابلیت های مهم این شبیه ساز شیء گرا بودن آن است که روند شبیه سازی را برای طراح بسیار ساده و قابل درک می کند. همچنین یکی دیگر از قابلیتهای آن سلسله مراتبی بودن روند طرح ریزی مسئله در آن است. به نحوی که بعدا خواهیم دید برای ایجاد یک سناریو و انجام یک شبیه سازی در آن، در سه سطح مختلف مسئله را می توان مطرح کرد.

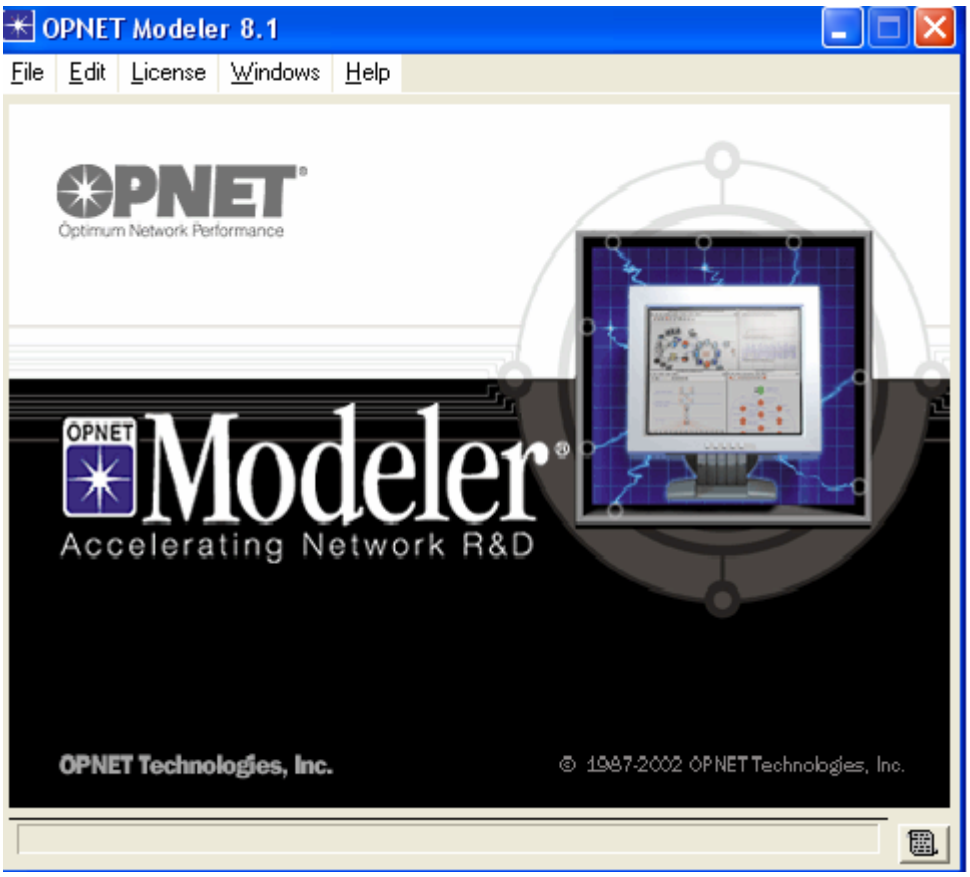
۲- معرفی شبیه ساز **OPNET**

---

[۱] - Hybrid
[۲] - Optimized Network Engineering Tool
[۳] - Network simulator
[۴] - Virtual InterNetwork Testbed
[۵] - Open source

OPNET یک شبیه ساز تجاری بوده که برای شبیه سازی پروتکلهای رایج در شبکه و مدلهای مختلف در شبکه های با رنج بسیار متنوع به کار می رود. به طوریکه در این نرم افزار مدلهایی برای شبیه سازی شبکه هایی از ابعاد کوچک و در حد یک **LAN** ساده تا شبکه های ماهواه ای بزرگ وجود دارد. در این نرم افزار از شبیه سازی رویدادهای گسسته جهت آنالیز عملکرد و رفتار شبکه ها استفاده می شود.

این نرم افزار بر روی سیستم عاملهای **Windows** و **Solaris** قابل نصب است. همچنین برای اجرای صحیح شبیه سازی بایدکامپایلر + **C+** یا **ANSI** نصب شده باشد. پس از نصب کامل نرم افزار که شامل قسمتهای **Modeler** , **Model library**, **documentation** است و اجرای اولیه آن جهت وارد کردن اطلاعات مربوط به **license** نرم افزار، می توان کار را با **OPNET** آغاز کرد. در این حالت پس از اجرای نرم افزار پنجره اصلی آن به صورت زیر باید نمایان شود.

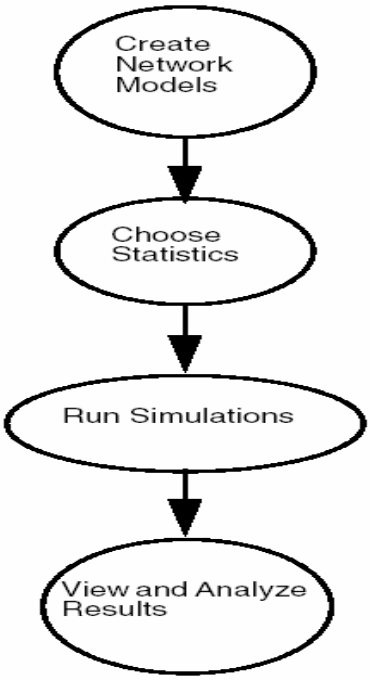


شکل (۱)

اگر بخواهیم توصیفی کلی از این شبیه ساز توانا داشته باشیم، موارد زیر را می توان عنوان کرد:

الف- این شبیه ساز ابزار های پرقدرتی را در جهت کمک به کاربر در انجام ۴ مرحله مختلف یک شبیه سازی فراهم می کند.

به طور کلی برای انجام یک شبیه سازی، می توان ۴ مرحله کاری مختلف در نظر گرفت. این مراحل در شکل ۲ به خوبی نشان داده شده است. با استفاده از این شبیه ساز می توان مراحل نشان داده شده در شکل را به خوبی اجراء کرد.



شکل(۲)

ب- در شبیه ساز **OPNET** یک ساختار سلسله مراتبی برای شبیه سازی به کار می رود. به عبارت دیگر مساله در سطوح مختلف تعریف می شود که در هر سطح جنبه های مختلف مدل مطرح می شود.

پ- کتابخانه این شبیه ساز بسیار کامل بوده و شامل آخرین پروتکلها و مدلهای ارائه شده در زمینه شبکه است. همچنین این قابلیت را نیز دارا می باشد که پژوهشگران شبکه برای خود اقدام به به روز رسانی مدلهای موجود در آن کنند.

۳- ساختار سلسله مراتبی در شبیه ساز OPNET

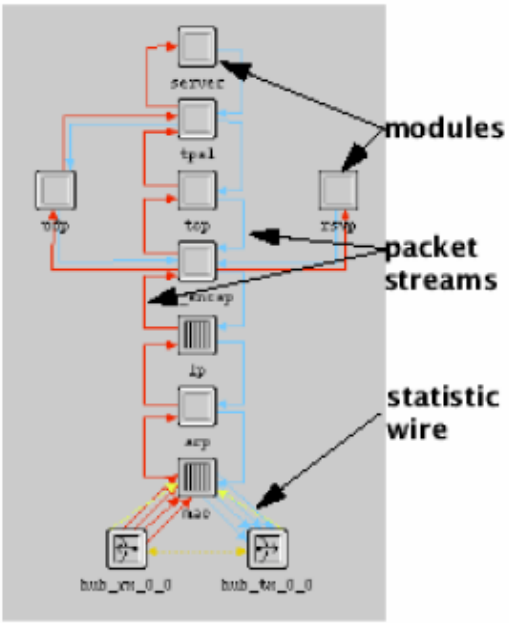
همانطور که قبلا عنوان شد روند طرح ریزی مساله به صورت سلسله مراتبی می باشد. به عبارت دیگر محیط شبیه سازی در آن دارای سه سطح طراحی است که عبارتند از:

I) **Network** یا **Project**

II) **Node**

III) **Process**

هر یک از این سه ابزار طراحی یک ویرایشگر[۱] نامیده می شوند و در هر کدام چنانچه خواهیم دید محیطی متفاوت برای طراحی یک بخش از مساله آماده شده است. در سطح اول طراحی، توپولوژی کلی شبکه شامل گره ها، لینکها و... رسم می شود. در سطح **Node** رفتار تک تک عناصر شبکه طراحی و ترسیم می شود. لذا برای هر عنصر در شبکه یک ساختار جداگانه بر مبنای ویژگیهای آن عنصر، در سطح **Node** تعریف می شود. در این ویرایشگر ساختار لایه ای هر عنصر شبکه و ارتباط میان لایه ها تعیین می شود. در این سطح طراحی، با ماژولها و دنباله بسته ها[۲] سر و کار داریم (شکل ۳) .



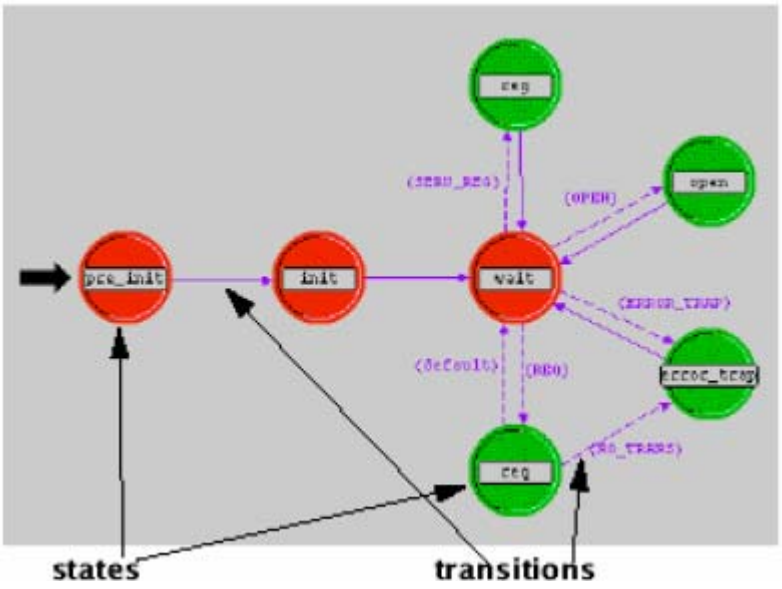
شکل (۳)

در سطح **process** هم نهایتا عملکرد و رفتار هر ماژول را می توان تعریف کرد. در واقع وظیفه ای که هر ماژول به عهده دارد را اصطلاحا **process** گویند. هر **process** مجموعه ای از دستورالعملها بوده که با استفاده از روش ماشین با حالت محدود[۳] (**FSM**) مدل می شود. در این سطح، شبیه سازی بر مبنای زبان برنامه نویسی **Porto-C** صورت می گیرد. **Porto-C** برای مدل کردن پروتکلها و

---

[۱] - Editor
[۲] - Packet Stream
[۳] - Finite State Machine

الگوریتمها طراحی شده است و شامل یک محیط گرافیکی از دیاگرامهای **STD**[1] ، کتابخانه ای از دستورات سطح بالا که به **KPs**[2] معروفند، و زبان برنامه نویسی **C** یا **C++** است. هر **STD** از مجموعه ای از حالتها تشکیل شده که بر حسب رخداد شرایط خاص گذار بین این حالتها صورت می گیرد شکل (۴).



شکل (۴)

البته به جز ویرایشگرهای اصلی ذکر شده، محیطهای ویرایش دیگری نیز در **OPNET** وجود دارند. این ویرایشگرها به نوعی در خدمت سه ویرایشگر اصلی عنوان شده بوده و مدلهایی ایجاد می کنند که در ویرایشگرهای اصلی مورد استفاده قرار می گیرند. در شکل (۵) ساختار سلسله مراتبی شبیه سازی در **OPNET** به خوبی نشان داده شده است.

در بخشهای بعدی هر یک از این محیطها را به طور جداگانه بررسی می کنیم.

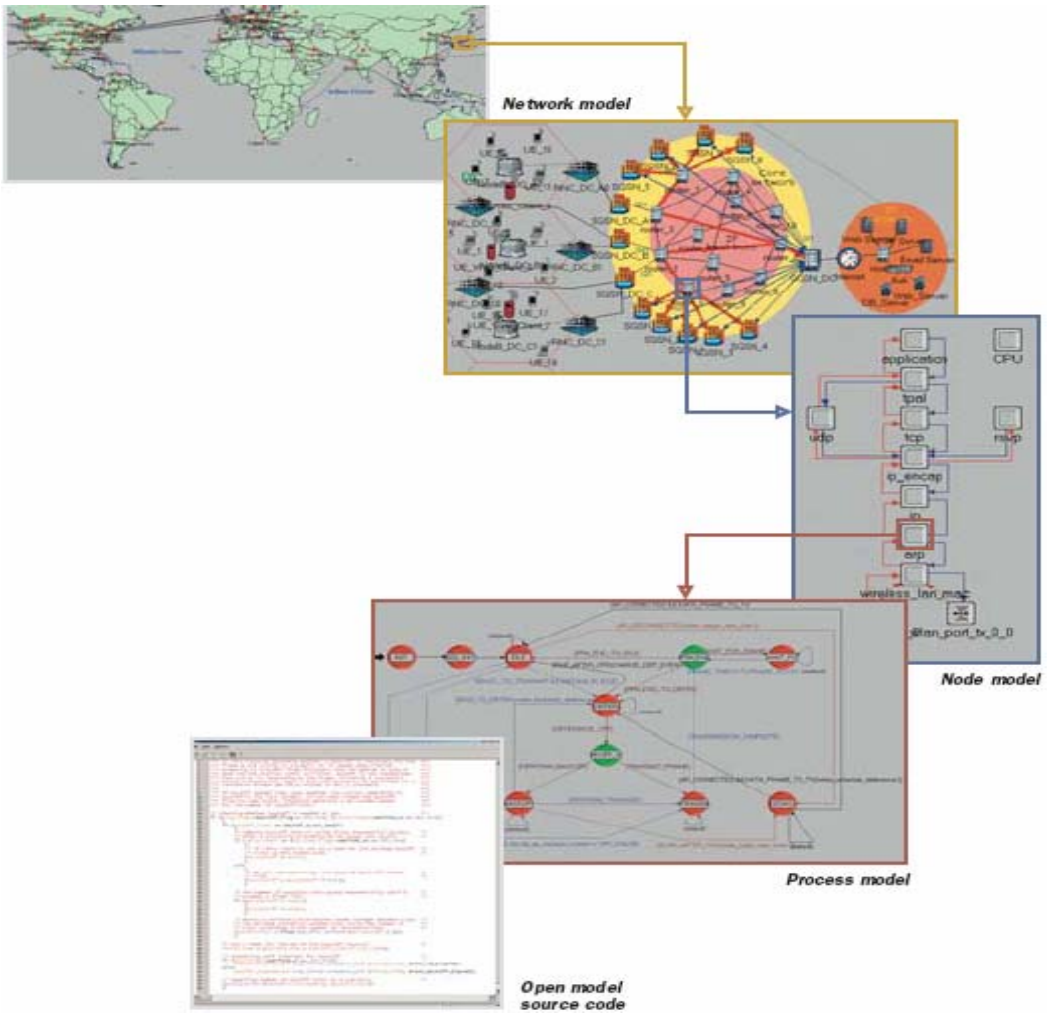
۳-۱- ویرایشگر شبکه (**Network editor or Project editor**)

این ویرایشگر محیط اولین مرحله از شبیه سازی است. در این ویرایشگر توپولوژی کلی شبکه با استفاده از مدلهایی از جدید ترین ادوات و تجهیزات مخابراتی و شبکه که در کتابخانه این نرم افزار موجود است، طراحی و مدل می شود. ساختار شبکه در این محیط شامل گره ها و لینکهایی است که این گره ها را به هم متصل می کند. هر یک از این گره ها دارای پارامترهای مختلف بوده که قابل تغییر

---

[1] - State Transition Diagrams
[2] - Kernel Procedures

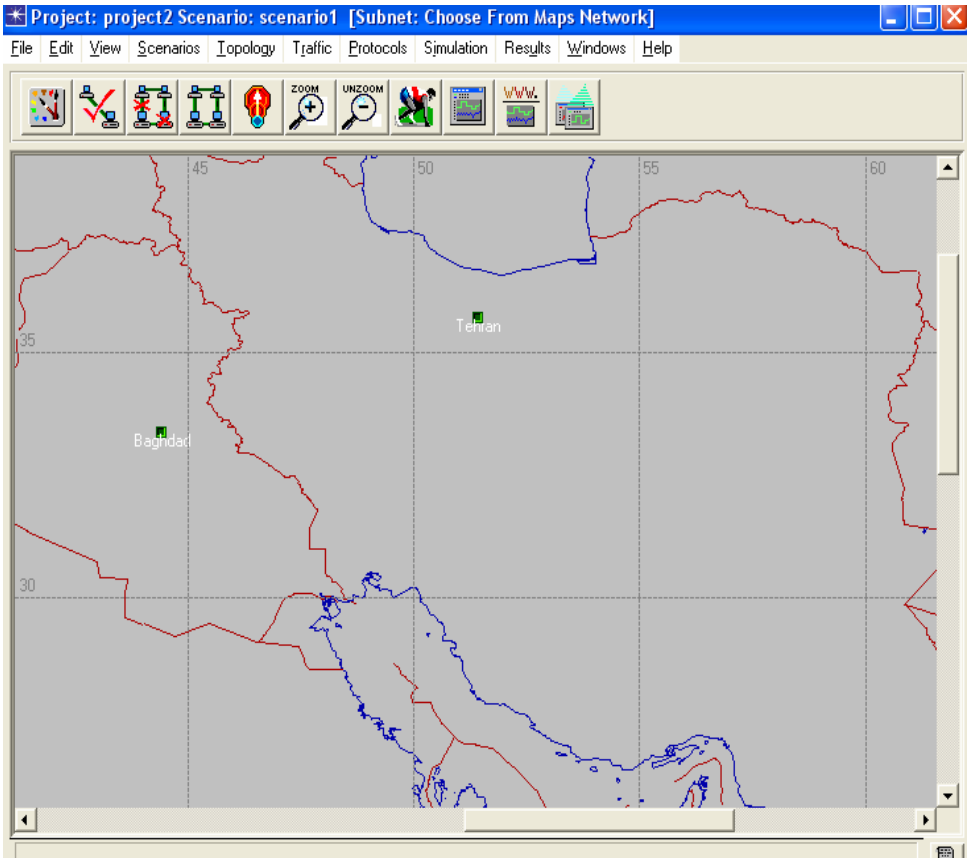
به نحو دلخواه هستند. برای رسم توپولوژی شبکه مورد نظر همانطور کـه در ادامـه توضـیح خـواهیم داد می توان گره ها و لینکهای مورد نظـر را از پنجـره ای کـه **object palette** نامیـده مـی شـود بـه روش کشیدن و



شکل (۵)

رها کردن[۱] به محیط ویرایشگر شبکه وارد کرد. همچنین برای رسم شـبکه هـای بـا ابعـاد وسـیع در ایـن محیط می توان موقعیت جغرافیایی شبکه را نیز به دلخواه تعیین کرد شکل(۶).

[۱] - Drag & Drop



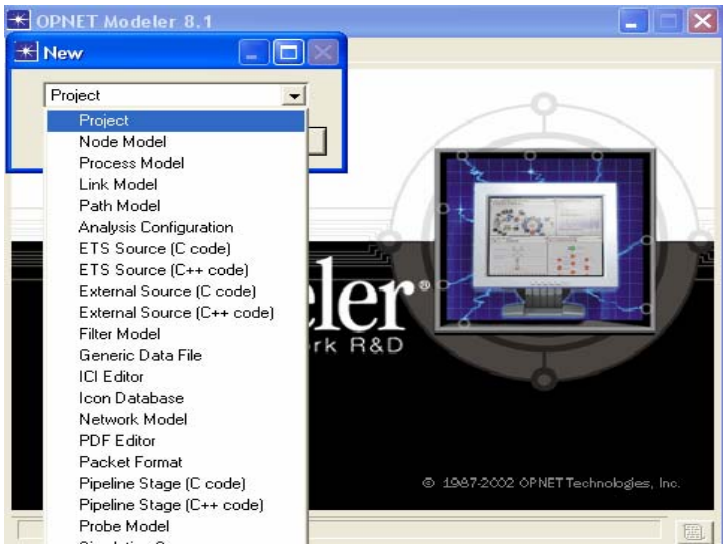
شکل (۶)

برای رسم ساختار شبکه های با تعداد گره های زیاد و ساختاری مشخص نظیر **bus**... **ring** , **star**, می توان روش سریعتری به کار برد که در ادامه به آن خواهیم پرداخت. در این محیط می توان پس از رسم ساختار کلی شبکه و تعیین پارمترهای موردنظر برای شبیه سازی اقدام به اجرای شبیه سازی و مشاهده نتایج حاصل از آن کرد.

برای وارد شدن به محیط ویرایشگر شبکه، پس از اجرای نرم افزار و باز شدن پنجره اصلی (شکل ۱) از منوی **File** یکی از گزینه های **New** یا **Open** را انتخاب می کنیم. با انتخاب گزینه **Open** می توان مدلهایی که در کتابخانه **OPNET** موجود است را باز کرد. این مدلها شامل اکثر پروتکلهای رایج شبکه نظیر **TCP**, **IP** , **RIP**, **Ethernet**, **ATM** و ... است. با باز کردن یکی از این مدلها، محیط ویرایشگر شبکه که در آن توپولوژی مربوط به استاندارد مربوطه رسم شده است نمایان می شود.

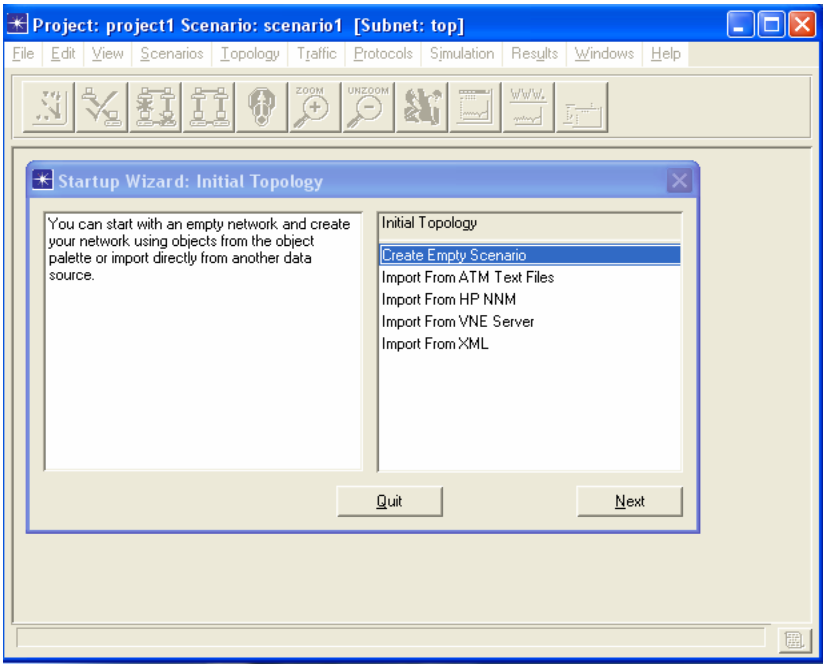
با انتخاب گزینه **New** می توان یک سناریوی جدید که مد نظر است ایجاد کنیم که قبل از باز شدن پنجره مربوط به ویرایشگر شبکه باید پارامترهای مربوط به نوع شبکه مورد نظر را انتخاب و تعیین کرد که در اینجا بیشتر در مورد آن صحبت خواهیم کرد. پس از انتخاب گزینه **New** از منوی **File** مطابق شکل (۷) پنجره کوچکی باز شده که در آن باید با توجه به اینکه در چه سطحی قصد انجام شبیه سازی

داریم (**Network**, **Node**, **Process**) ، یکی از گزینه های مربوطه را انتخاب کرد. در اینجا چون ما در سطح (**Network**) **project** کار می کنیم، گزینه **project** را انتخاب می کنیم. بعد از تایید آن و انتخاب نام برای پروژه مورد نظر، پنجره ای باز خواهد شد که در آن باید توپولوژی اولیه شبکه را تعیین کرد.



شکل (۷)

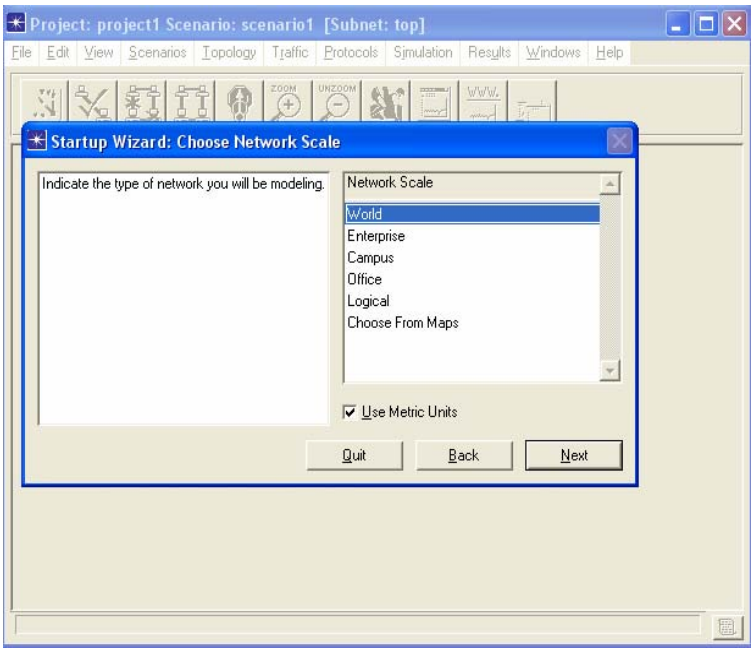
همانطور که در ادامه توضیح خواهیم داد، در صورتی که بخواهیم ساختار شبکه بر مبنای یک ساختار قبلی که در قالب یک فایل خاص (**XML** یا فایل متنی **ATM** یا فایلهای تعریف شده دیگر) قرار دارد می توان در این پنجره گزینه های مربوط به هر فایل را انتخاب کرد (شکل ۸) . در غیر این صورت با انتخاب گزینه اول، می توان شبکه را از ابتدا و با روشهای موجود در ویرایشگر شبکه طراحی کرد.



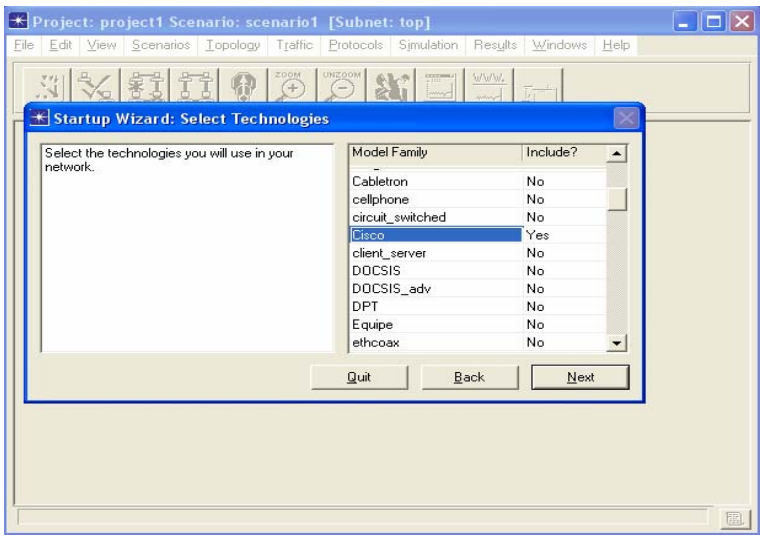
شکل (۸)

برای این منظور پنجره ای دیگر ظاهر خواهد شد که در آن باید مشخصات جغرافیایی شبکه را انتخاب کرد (شکل ۹) . در اینجا بسته به نوع طراحی می توان یکی از گزینه های مورد نظر را انتخاب کرد. در صورت انتخاب بعضی از این گزینه ها باید در مرحله بعدی ابعاد جغرافیایی شبکه را بر حسب متر یا کیلومتر در پنجره ای که بعد از آن ظاهر خواهد شد وارد کرد. همچنین با انتخاب گزینه های **world** یا **choose from maps** می توان موقعیت شبکه را با توجه به نقشه کره زمین انتخاب کرد.

بعد از تعیین موقعیت جغرافیایی شبکه، پنجره ای ظاهر خواهد شد که در آن می توان تکنولوژی ادوات مورد استفاده برای شبیه سازی را انتخاب کرد (شکل ۱۰)



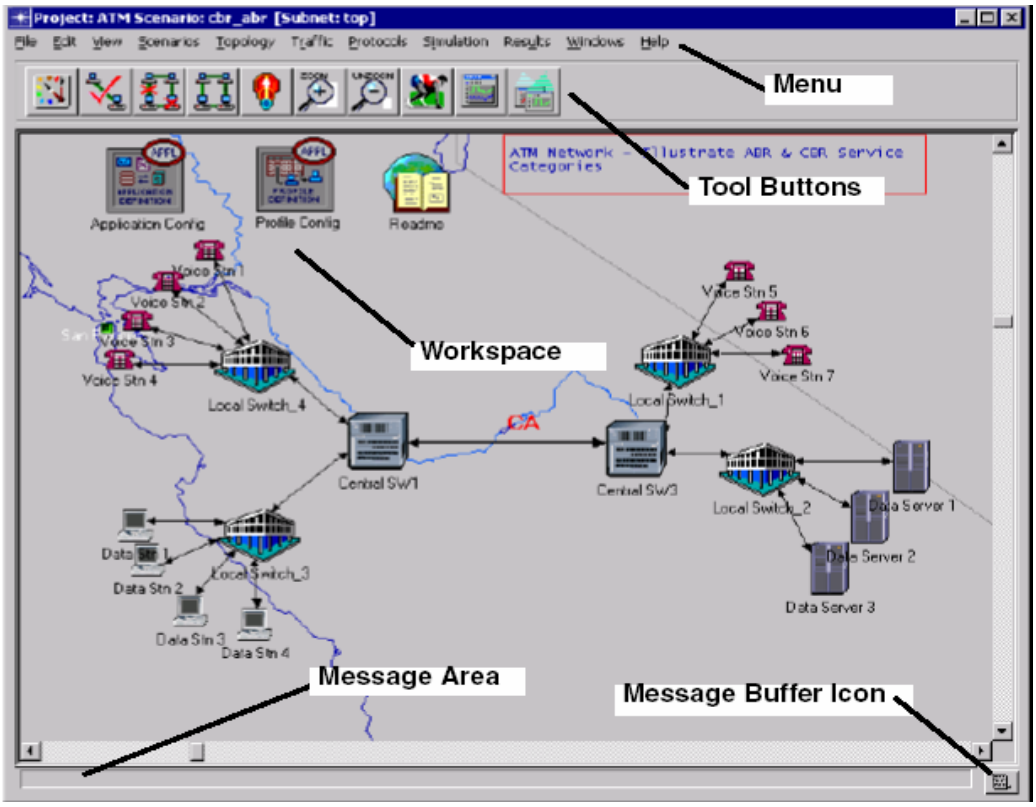
شکل (۹)



شکل (۱۰)

آخرین پنجره ای که برای وارد کردن اطلاعات مورد نظر شبکه ظاهر خواهد شد، پنجره ای است که در آن تمامی موارد انتخاب شده توسط کاربر در پنجره های قبلی (ساختار جغرافیایی شبکه، تکنولوژی ادوات مورد استفاده و...) به صورت خلاصه جهت تایید نهایی کاربر وجود دارد. بعد از تایید آن مراحل اولیه جهت کار با ویرایشگر شبکه تمام شده و محیط این ویرایشگر ظاهر خواهد شد که می توان طراحی را بر اساس پارامترهای مورد نظر انجام داد.

به عنوان مثال در شکل زیر محیط این ویرایشگر را برای یک سناریو که بر مبنای استاندارد **ATM** طرح شده است می بینیم.



شکل(۱۱)

در این شکل همچنین نواحی مختلف این محیط نشان داده شده است که در اینجا به توضیح آنها می پردازیم:

الف- نوار فهرست (**Menu Bar**) :

این ناحیه که در بالای پنجره واقع است شامل منوهای مختلف نظیر **File**, **Edit**, **View**, ... است که کاربرد اکثر این منوها احتیاج به توضیح خاصی ندارد. البته بعضی از این منوها کاربردهای خاص خود را داشته که در موقع لزوم به آنها اشاره می کنیم. راه دیگر دستیابی به عملکرد این منوها کلیک راست

کردن بر روی اشیاء (گره ها و لینکها) و یا در محیط خالی پنجره است که دستیابی به منوها از این طریق نیز امکان پذیر است.

ب- محیط کار (Workspace)

بخش مرکزی پنجره این ویرایشگر که مدل شبکه در آن رسم می شود را **Workspace** می گویند. با کلیک راست کردن در فضای خالی این محیط می توان به منوهای موجود در نوار فهرست دسترسی پیدا کرد.

ج- ناحیه پیغام (Message Area)

این ناحیه در پایین پنجره واقع شده و در آن پیغامهای مورد نیاز برای اطلاع کاربر ظاهر می شوند شکل(۱۲) .

شکل (۱۲)

همچنین در سمت راست این ناحیه آیکنی قرار دارد که به آن **Message Buffer Icon** می گویند. با کلیک کردن بر روی این آیکن پنجره ای به نام بافر پیغام (**Message Buffer**) باز می شود. کاربرد این پنجره در مواردی است که طول پیغام زیاد بوده و فقط بخشی از پیام در **Message Area** قابل رویت است. در این حالت می توان کل پیغام را در این پنجره مشاهده کرد. همچنین کلیه پیامهای قبلی نیز در این پنجره ذخیره و موجود است.

د- دکمه های ابزار (Tool Buttons) :

در این ناحیه عملکردهایی از این ویرایشگر که مهمتر از بقیه بوده و به وفور استفاده می شوند جهت در دسترس بودن قرار داده شده اند (شکل۱۳). همچنین تمامی این عملکردها از طریق یکی از منوهای موجود در **Menu Bar** نیز قابل دسترسی هستند.

در شکل زیر این ناحیه را می توان مشاهده کرد که در آن هر گزینه با یک شماره مشخص شده است و در جدول (۱) عملکرد هر گزینه به صورت خلاصه آمده است.

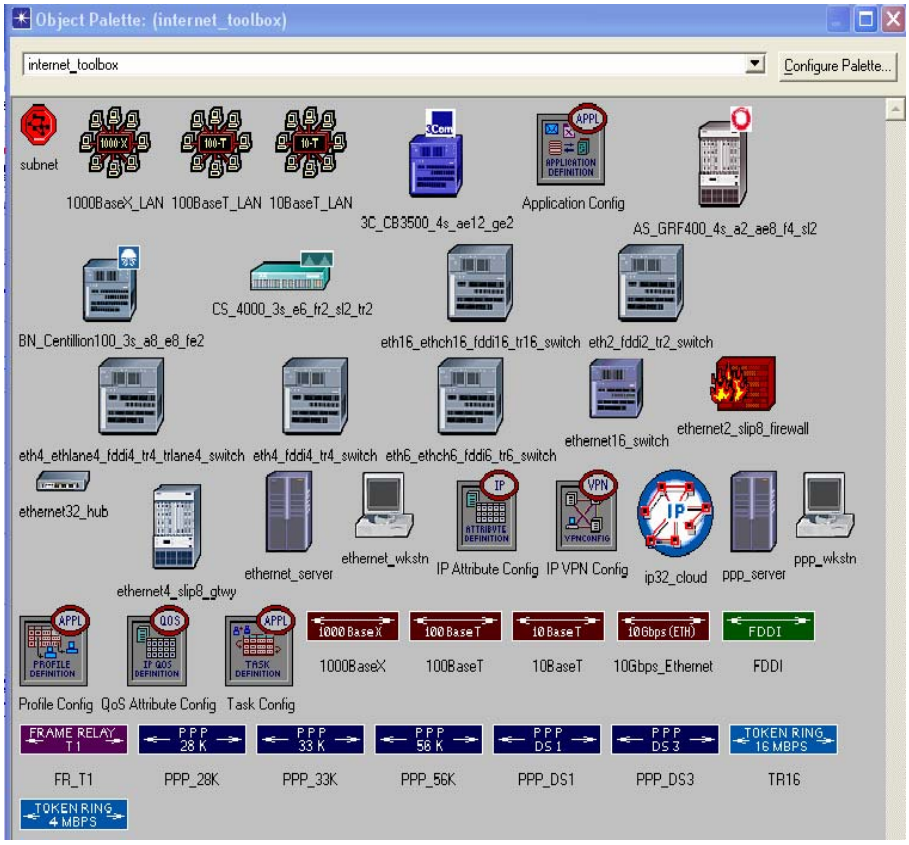
شکل(۱۳)

جدول (۱)

1	Open object palette	6	Zoom
2	Check link consistency	7	Restore
3	Fail selected objects	8	Configure discrete event simulation
4	Recover selected objects	9	View simulation results
5	Return to parent subnet	10	Hide or show all graphs

۳-۱-۱) پنجره **object palette**

با استفاده از گزینه شماره ۱ می توان پنجره **object palette** باز کرد. در این پنجره همانطور که قبلا هم عنوان شد کلیه تجهیزات مورد استفاده در طراحی سخت افزاری شبکه شامل روتر، سوئیچ، سرور و همچنین لینکهای مختلف وجود دارد و می توان از آن به روش کشیدن و رها کردن این ادوات را به محیط شبیه سازی وارد کرد. در این پنجره این قابلیت هم وجود دارد که تکنولوژی ساخت این ادوات را نیز به دلخواه انتخاب کرد. شکل (۱۴) پنجره **object palette** را نشان می دهد.

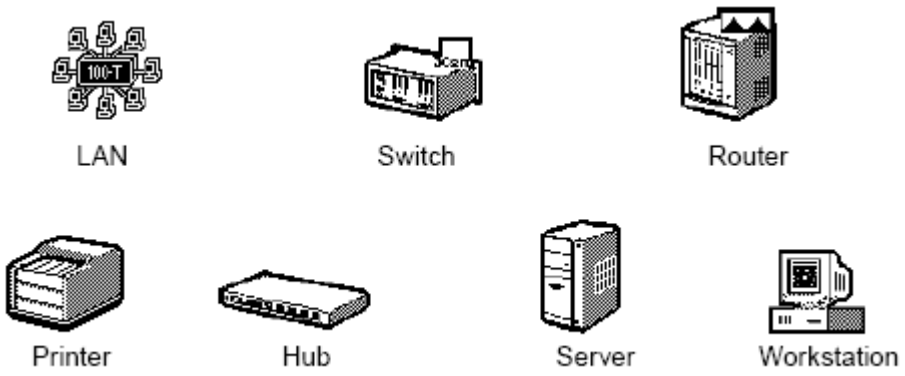


شکل (۱۴)

به طور کلی ادوات موجود در پنجره **object palette** را می توان به ۴ دسته زیر تقسیم کرد که در ادامه آنها را توضیح می دهیم:

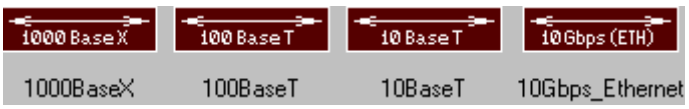
۱- devices -۲- links -۳- LANs and clouds -۴- utility objects

device ها اکثریت ادوات موجود در این پنجره را تشکیل می دهند و ادوات سخت افزاری نظیر روترها، سوئیچها، **hub** ، **server** ، **Firewall** هستند (شکل ۱۵) .



شکل (۱۵)

لینکها جهت اتصال گره ها به یکدیگر در تکمیل ساختار شبکه به کار می روند. هر لینک پارامترهایی نظیر سرعت ارسال اطلاعات، تاخیر، احتمال خطا و... داشته که در **OPNET** نیز این پارامترها برای هر لینک قابل تعیین است. یکی از مهمترین ویژگی های یک لینک سرعت ارسال اطلاعات در آن است که بر حسب بیت بر ثانیه بیان می شود. در **OPNET** این ویژگی به صورت صریح در شکل گرافیکی یک لینک نشان داده شده است و در واقع هر لینک با این پارامتر شناسایی می شود. به عنوان مثال یک لینک که با **۱۰BaseT** نشان داده می شود سرعت **Mbps** ۱۰ دارد. در شکل (۱۶) مدل یک لینک در **OPNET** نشان داده شده است.



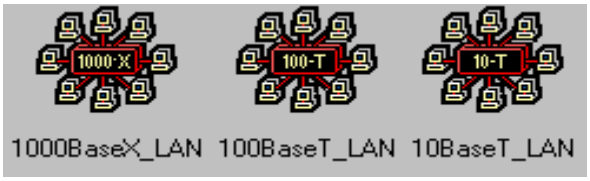
شکل (۱۶)

در **OPNET** این امکان فراهم شده است که هر گره پایانی[۱] همراه با جزئیات آن قابل شبیه سازی و مدل کردن است. اما در بعضی موارد کاربر ترجیح می دهد که یک شبکه پایانی کوچک (یا یک **LAN** ساده) به صورت خلاصه در با یک گره که مدل کننده تمام ویژگیهای آن است نشان داده شود.

[۱] - End System

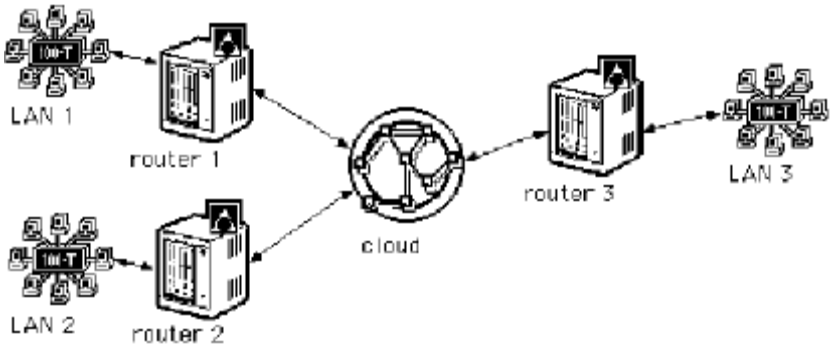
با اینکار هم کاربر از درگیری با جزئیات درون یک **LAN** رهایی می یابد و هم اینکه در پیکربندی این **LAN** زمان کمتری نسبت به حالتی که بخواهیم تک تک ادوات درون این **LAN** را پیکر بندی کنیم صرف می شود. با استفاده از مدلهای **LAN** می توان به راحتی تعداد زیادی کاربر را برای انجام شبیه سازی وارد شبکه کرد.

در شکل (۱۷) چند مدل **LAN** موجود در پنجره **objective palette** نشان داده شده است.



شکل (۱۷)

مدلهای دیگری که جهت ساده سازی ساختار شبکه در **OPNET** فراهم شده است، ابرها[۱] هستند. ابرها بخشی از یک شبکه **WAN** هستند که جهت ساده سازی به صورت یک گره در پنجره **object palette** قرار دارند. بیشترین کاربرد ابرها در مدل کردن اتصالات ستون فقرات[۲] شبکه های **WAN** است. ابرهای **IP** , **ATM** و **Frame – relay** مهمترین مدلهای موجود در کتابخانه **OPNET** هستند. در شکل زیر یک شبکه **WAN** که در آن یک ابر جهت ساده سازی ساختار شبکه به کار رفته است نشان داده شده است.



شکل (۱۸)

در رسم توپولوژی یک شبکه در **OPNET** بلوکهایی به کار برده می شوند که در دنیای واقعی به صورت یک واحد سخت افزاری وجود ندارند. به این بلوکها که هر یک در پنجره **objective palette** قرار دارند اصطلاحا اشیاء سودمند (**utility object**) گفته می شود. در حالت کلی هر یک از

---

[۱] - Clouds
[۲] -Backbone

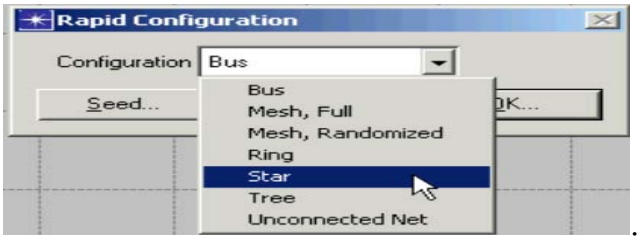
این بلوکها در شبکه یک وظیفه جانبی بر عهده دارد. به عنوان مثال وظیفه پیکر بندی منابع شبکه می تواند بر عهده یکی از این بلوکها گذاشته شود.

۳-۱-۲) رسم ساختار شبکه در محیط ویرایشگر شبکه

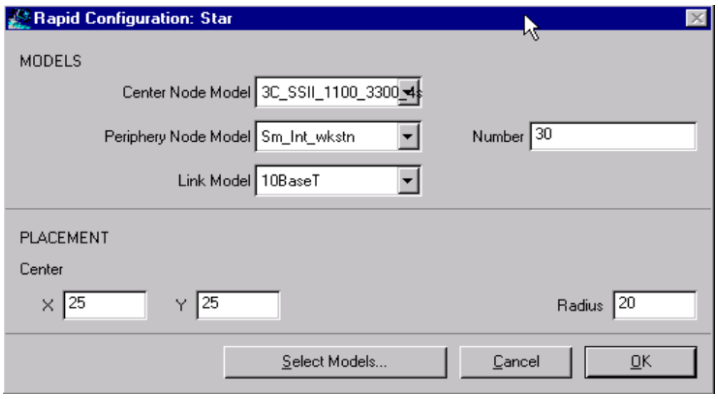
به طور کلی برای رسم توپولوژی یک شبکه در **OPNET** سه روش وجود دارد:

در روش اول می توان هر گره یا لینک مورد نیاز برای طراحی شبکه را به طور جداگانه از پنجره **object palette** به محیط ویرایشگر وارد کرد. برای این کار ابتدا با کلیک چپ کردن بر روی وسیله مورد نظر در پنجره **object palette** می توان آن را به محیط ویرایشگر منتقل کرد. با چندین بار کلیک چپ کردن در این محیط می توان به تعداد دلخواه از وسیله مورد نظر به محیط شبیه سازی انتقال داد. در نهایت برای خاتمه انتقال وسیله مورد نظر باید در محیط ویرایشگر کلیک راست کرد تا فرایند انتقال آن وسیله به پایان برسد.

یکی از معایب این روش در رسم ساختار شبکه این است که در مواردی که تعداد گره ها و لینکهای شبکه زیاد است زمان زیادی برای کامل شدن ساختار شبکه طول می کشد. ضمن اینکه ساختار شبکه نیز ممکن است بسیار پیچیده بوده و لذا برای جلوگیری از وقوع اشتباه در نحوه اتصالات حوصله زیادی باید به خرج داد. برای این منظور روش دومی برای رسم ساختار کلی شبکه پیشنهاد می شود که استفاده از گزینه **Rapid Configuration** است. با استفاده از این گزینه می توان در مواردی که شبکه مورد نظر دارای ساختاری منظم (مثل **star**, **tree**, **bus**, ...) و با تعداد گره های زیاد است، در کمترین زمان ممکن ساختار شبکه را رسم کرد. برای این منظور می توان از منوی **Topology** گزینه **Rapid Configuration** را انتخاب کرد. بعد از این کار پنجره ای مطابق شکل (۱۹) زیر باز خواهد شد. پس از انتخاب یکی از گزینه های مورد نظر بر حسب ساختاری که برای شبکه در نظر داریم و تایید آن، در مرحله بعدی باید پارامترهای خاص شبکه نظیر تعداد گره ها، نوع گره ها و لینکها، و همچنین اطلاعاتی راجع به موقعیت مکانی گره ها، را وارد کرد. به عنوان مثال اگر ساختار شبکه را **star** انتخاب کنیم، پنجره ای مطابق شکل (۲۰) باز خواهد شد که می توان اطلاعات مربوطه را در آن وارد کرد.



شکل (۱۹)



شکل (۲۰)

سومین روش برای رسم ساختار شبکه در محیط ویرایشگر شبکه، وارد کردن توپولوژی شبکه با استفاده از یک فایل خارجی است. در **OPNET** این قابلیت ایجاد شده است که بتوان از یک فایل خارجی اطلاعات مربوط به ساختار شبکه را وارد کرد. به طور کلی فایلهایی که برای این منظور به کار می روند عبارتند از:

فایلهای **XML** ، فایلهای متنی **ATM** (**ATM text files**) ، **Device configuration data** و ... .

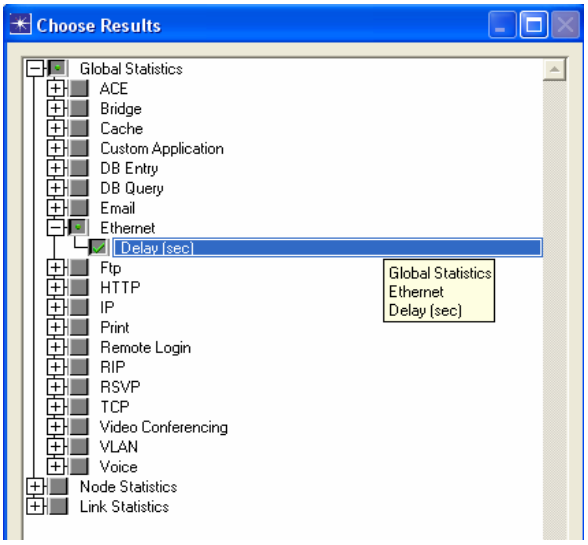
که برای این منظور بیشتر از فایلهای **XML** استفاده می شود[۳].

۳-۱-۳) مشخص کردن پارامترهای موردنظر برای اجرای شبیه سازی

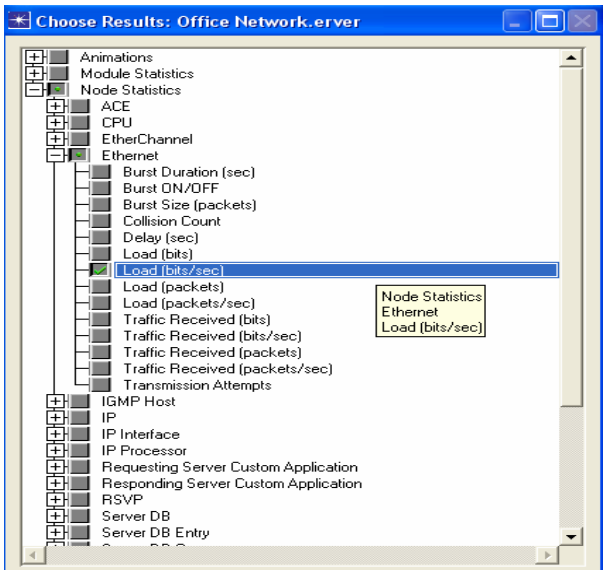
بعد از رسم ساختار شبکه در محیط ویرایشگر نوبت به اجرای شبیه سازی می رسد. برای این منظور باید ابتدا پارامتر یا پارامترهایی که می خواهیم تغییرات آنها را در شبکه بررسی کنیم تعیین کنیم. به عنوان مثال اگر در یک مسئله با توسعه یک شبکه کوچک مواجه باشیم باید دید آیا با زیاد کردن بار شبکه در اثر گسترش آن سرور عملکرد مورد قبولی خواهد داشت و یا خیر و همچنین تاخیری که در اثر توسعه شبکه تحمیل می شود قابل قبول خواهد بود. نرم افزار **OPNET** این قابلیت را دارد که به طور همزمان نتایج حاصل از شبیه سازی قبل و بعد از یک رویداد مختلف را بیان کرده و با یکدیگر مقایسه کند.

در **OPNET** پارامترهایی که قصد مشاهده میزان تغییرات آنها در شبکه را داریم به دو دسته تقسیم می شوند (به این پارامترها اصطلاحا **statistics** گفته می شود) . دسته اول عمومی بوده (**global**) و در مورد همه ادوات شبکه صدق می کند. در مثال عنوان شده میزان تاخیر ناشی از توسعه شبکه پارامتری از این دسته است. چرا که همه کاربران شبکه در آن لحاظ می شوند. اما دسته دوم پارامترهایی هستند که مربوط به یک وسیله خاص بوده (**object statistics**) و فقط باید اطلاعات از آن جمع آوری شده و نتایج گزارش شود. در مثال فوق اگر بخواهیم میزان بار تحمیل شده به سرور را قبل و بعد از توسعه شبکه بررسی کنیم این پارامتر فقط مربوط به سرور بوده و لذا در دسته دوم این پارامترها قرار می گیرد. در هر صورت برای تعیین و تنظیم این پارامترها می توان با انتخاب گزینه **Choose individual statistics** از منوی **Simulation** اقدام به تعیین این پارامترها کرد. روش دیگر این است که بر روی

وسیله مورد نظر در محیط ویرایشگر کلیک راست کرده و گزینه **Choose individual statistics** را انتخاب کرد. البته این کار فقط برای تعیین پارامترهای مخصوص یک وسیله (دسته دوم پارامترها در تقسیم بندی) صورت می گیرد. برای پارامترهای عمومی می توان در محیط خالی ویرایشگر کلیک راست کرده و همان گزینه را انتخاب کرد. با انجام این کار پنجره ای مشابه شکل (۲۱) برای پارمترهای دسته اول و مشابه شکل (۲۲) برای پارامترهای دسته دوم باز خواهد شد.



شکل(۲۱)



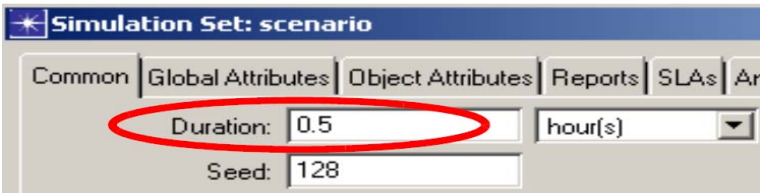
شکل (۲۲)

بعد از اینکه پارامترهای مربوطه تعیین شدند، نرم افزار آماده اجرای شبیه سازی است. برای این کار می توان از منوی **Simulation** گزینه **Run simulation** را انتخاب کرد و یا با کلیک کردن بر روی دکمه **configure/run simulation** این کار را انجام داد (شکل ۲۳) . بعد از این کار پنجره ای

مطابق شکل (۲۴) باز شده که در آن باید تنظیمات جانبی شبیه سازی نظیر مدت زمان شبیه سازی رفتار شبکه را انجام داد. با انتخاب گزینه **Run** در پایین پنجره، عملیات شبیه سازی آغاز می شود.

در طی زمانی که سیستم در حال اجرای شبیه سازی است پنجره ای باز می شود که میزان پیشرفت محاسبات را نشان می دهد.

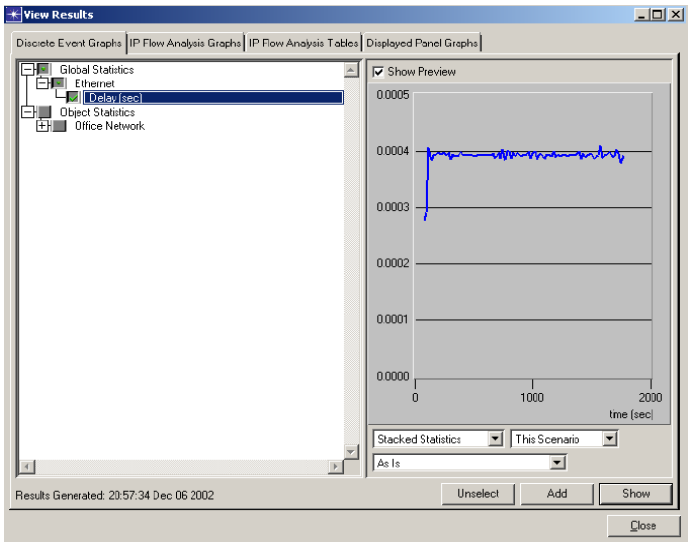
شکل (۲۳)



شکل (۲۴)

۳-۱-۴) مشاهده نتایج شبیه سازی

برای مشاهده نتایج شبیه سازی در **OPNET** راههای مختلفی وجود دارد. یکی از این راهها از طریق انتخاب گزینه **View Results** است. با کلیک راست کردن در محیط خالی ویرایشگر و یا بر روی وسیله مورد نظر (بسته به اینکه نتایج مربوط به چه پارامتری را بخواهیم مشاهده کنیم) می توان از منوی مربوطه این گزینه را انتخاب کرد. بعد ازانتخاب این گزینه مجددا مشابه مرحله قبلی باید پارامتر مورد نظر که به دنبال مشاهده نتایج آن هستیم را انتخاب کنیم (شکل ۲۵).



شکل (۲۵)

۳-۲) ویرایشگر Node

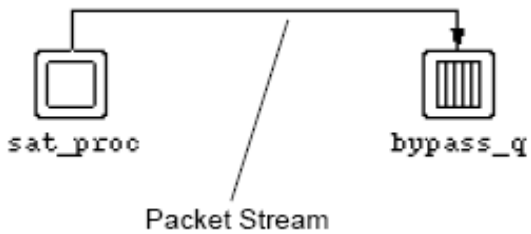
همانطور که قبلا هم عنوان شد، در این ویرایشگر ساختار لایه ای تک تک عناصر شبکه و نحوه ارتباط لایه ها با یکدیگر تعیین وتعریف می شود. لذا در این سطح با جریان داخلی بسته ها درون یک عنصر شبکه سر و کار داریم. به طور کلی هر عنصر شبکه که دارای تمامی یا برخی از قابلیتهای زیر باشد را یک گره تعریف می کنیم:

ایجاد بسته های اطلاعاتی، ذخیره آنها، ارسال و دریافت آنها و در نهایت مسیر یابی داخلی و انجام پردازشهای لازم نظیر آنالیز محتویات بسته ها، مالتی پلکسینگ، کشف و تصحیح خطاهای احتمالی بسته و... .

لذا هر گره شامل ادوات سخت افزاری لازم و همچنین نرم افزارهای موردنیاز برای انجام چنین پردازشهایی است.

در این ویرایشگر برای مدل کردن لایه های مختلف تعریف شده درون یک عنصر از شبکه، از بلوکهایی با عنوان ماژول استفاده می شود. هر ماژول وظیفه مخصوص به خود را در مدل کردن یکی از جنبه های عملکرد داخلی هر گره بر عهده دارد. به عنوان مثال ممکن است یک ماژول برای تولید بسته داشته باشیم و ماژول دیگری برای امر مسیر یابی و پردازش هدر بسته و همچنین ماژولی نیز مدل کننده واحد ارسال اطلاعات باشد. بنابراین یک مدل در سطح **Node** از چندین ماژول تشکیل شده که برای عناصر پیچیده شبکه، تا چند صد ماژول هم ممکن است داشته باشیم.

ماژولها در این ویرایشگر به وسیله دنباله بسته ها[1] و یا **statistic wires** به یکدیگر متصل می شوند. **Packet streams** اتصالاتی هستند که بسته های اطلاعاتی را از ماژول مبدا به ماژول مقصد منتقل می کنند. همچنین این اتصالات به صورت لینکهایی مطمئن و عاری از هر گونه خطا و دارای پهنای باند نامحدود (بطوریکه بسته ها با هر طولی بدون تاخیر به منتقل می شوند) در نظر گرفته می شوند.

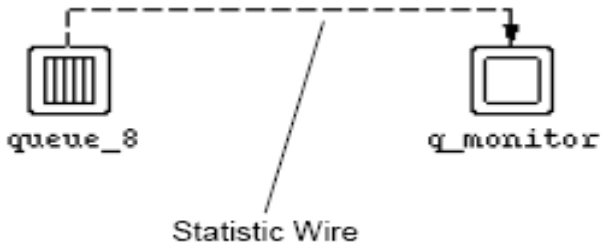


شکل (۲۶)

Statistic wires نیز مشابه **packet streams** اطلاعات را از یک ماژول به ماژول دیگر منتقل می کنند. اما بر خلاف packet streams که بسته های اطلاعاتی را حمل می کنند،**statistics wires**

[1]- Packet streams

بسته های کنترلی که باید بین ماژولها جهت انجام عملیات سیگنالینگ مبادله شوند را منتقل می کنند. همچنین با استفاده از **statistic wires** این قابلیت برای هر ماژول فراهم می شود که یک کمیت متغیر را در بین ماژولهای دیگر مونیتور کند.



شکل (۲۷)

در بخش بعدی به بررسی انواع ماژولهای موجود در این ویرایشگر می پردازیم.

۳-۲-۱) انواع ماژولها در ویرایشگر **Node**

به طور کلی در این ویرایشگر ۵ نوع ماژول مختلف می توان تعریف کرد که در اینجا به بررسی هر یک از آنها می پردازیم:

الف) ماژول پردازشگر[۱]

ماژولهای پردازشگر را می توان بلوکهای اولیه و پایه ای برای ایجاد هر مدل در این ویرایشگر دانست. این دسته از ماژولها برای انجام پردازشهای کلی بر روی بسته های اطلاعاتی به کار می روند. به عنوان نمونه یک ماژول پردازشگر می تواند یک بسته را از وردی خود دریافت کرده و آن را پس از انجام پردازشهای لازم به خروجی ارسال کند.

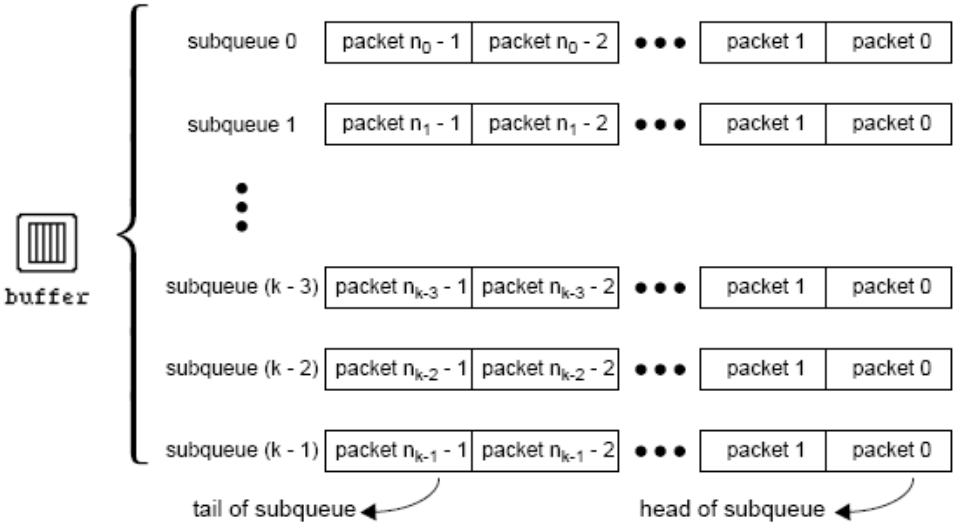
این ماژولها به منظور دریافت و ارسال بسته ها به وسیله انواع مختلفی از **packet streams** به ماژولهای دیگر متصل می شوند. به عنوان مثال یک ماژول پردازشگر که وظیفه انجام عملیات مالتی پلکسینگ چندین سیگنال بر روی یک لینک را بر عهده دارد، دارای چندین **packet streams** ورودی و یک **packet streams** خروجی است.

---

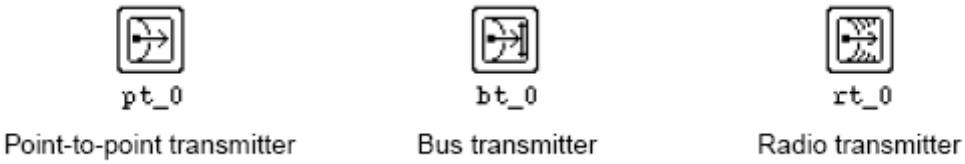
[۱] - Processor module

ب) ماژولهای صف[1]

هر ماژول صف از تعدادی زیر صف[2] تشکیل شده است که تعداد آنها را می توان به دلخواه تعیین کرد. هر زیر صف را می توان به طور جداگانه پیکربندی کرد. در ابتدا ظرفیت هر زیر صف به صورت پیش فرض نامحدود است، ولی می توان آن را به دلخواه تغییر داد. این ماژولها نیز مشابه ماژولهای پردازشگر از طریق **packet streams** به دیگر ماژولها متصل می شوند تا ارسال و دریافت بسته ها میسر شود. شکل زیر ساختار داخلی یک ماژول صف را با تعدادی زیر صف نشان می دهد.



پ) ماژولهای فرستنده



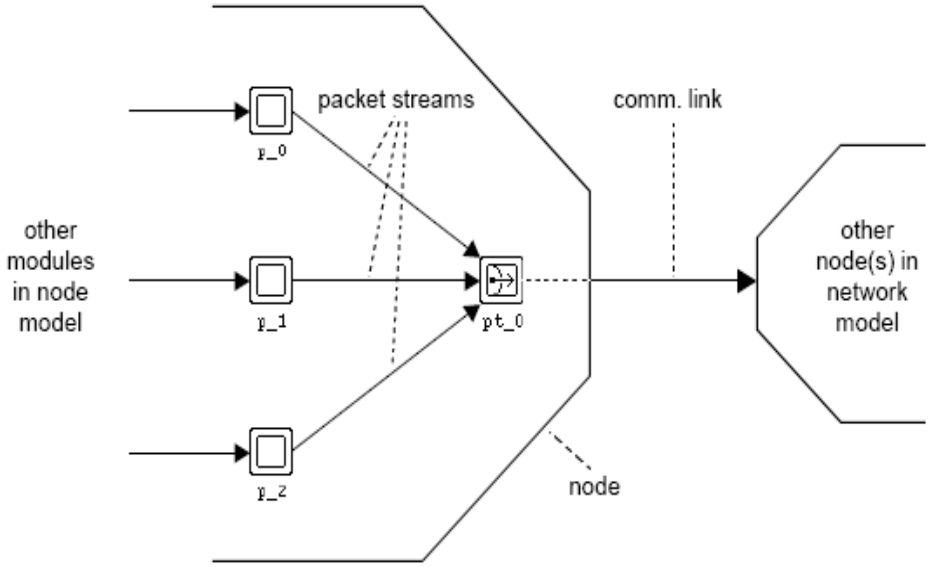
ماژولهای فرستنده را می توان رابط بین **packet streams** درون یک گره و لینک فیزیکی خارج آن گره دانست. در حالت کلی منوط به اینکه نوع لینک فیزیکی چه باشد سه نوع ماژول فرستنده مطابق شکل فوق تعریف می شود. اگرچه هر سه نوع از این ماژول رفتار مشابهی در برخورد با بسته های وارد

---

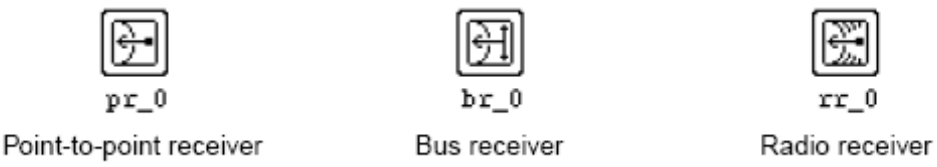
[2] - Queue modules
[2] - Subqueue

شده به آنها دارند، اما به دلیل متفاوت بودن ماهیت لینک فیزیکی که در هر کدام مدل می شود، این ماژولها با یکدیگر در برخی جنبه ها متفاوتند.

عملکرد هر ماژول فرستنده را می توان بدین نحو تشریح کرد که پس از دریافت بسته از یک یا چند **packet stream** ورودی خود، آنها را به نحو مناسب بر روی لینکهای مخابراتی مربوطه که به خروجی خود متصل اند ارسال می کنند. در واقع این ماژولها حکم همان لایه فیزیکی در ساختار لایه ای استاندارد را بر عهده دارند. شکل زیر نحوه اتصالات این ماژول را در یک مثال ساده نشان می دهد.

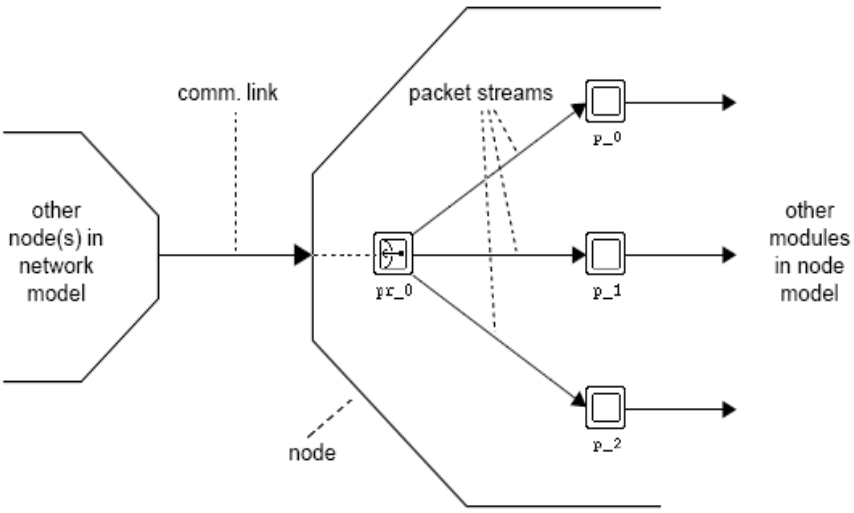


ت) ماژولهای گیرنده



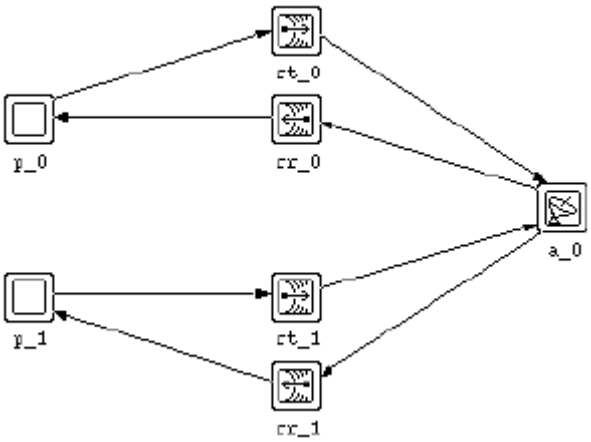
ماژولهای گیرنده را می توان رابط بین لینکهای فیزیکی و **packet streams** درون یک گره دانست و مشابه ماژولهای فرستنده بر حسب نوع لینک مخابراتی به کار رفته، بر سه نوع می باشند.

ماژولهای گیرنده می توانند بسته ها را بعد از دریافت از لینک بر روی یک یا چند packet stream خروجی خود ارسال کند. لذا در مدل یک گره در این ویرایشگر، ماژول گیرنده را می توان به عنوان منبع اطلاعات در نظر گرفت.



ث) ماژول آنتن

این ماژول که تنها در نسخه **OPNET Modeler / Radio** موجود است، برای تعیین ویژگیهای آنتن به کار رفته در ماژول فرستنده و یا گیرنده به کار می رود. با اینکار می توان به طور همزمان ویژگیهای یکسانی برای آنتنهای فرستنده و گیرنده موجود در یک گره تعریف کرد. به عنوان مثال با اینکار ماژولهای فرستنده و گیرنده می توانند از یک آنتن مشترک برای ارسال و دریافت اطلاعات استفاده کنند. شکل زیر این مطلب را به خوبی نشان می دهد.



در بخش بعدی به بررسی منوهای موجود در محیط ویرایشگر **Node** می پردازیم.

۳-۲-۲) منوهای مختلف در ویرایشگر **Node**

همانطور که قبلا هم عنوان شد، ویرایشگر Node قابلیتهای فراوانی را برای شبیه سازی عملکرد تک تک گره های یک شبکه در اختیار کاربر قرار می دهد. کاربر می تواند از طریق منوهای موجود در نوار فهرست[1] که در قسمت فوقانی محیط این ویرایشگر قرار گرفته است، به این قابلیتها دسترسی پیدا کند. در این بخش به بررسی این منوها می پردازیم:

- منوی فایل **(File)**

این منو شامل عملکردهای سطح بالا نظیر باز کردن و یا بستن یک پروژه شبیه سازی شده، ذخیره تغییرات انجام شده بر روی پروژه و... است.

- منوی **Edit**

شامل گزینه هایی **Undo**، **Cut**، **Copy**، **Paste** و ... بوده که کاربرد آنها احتیاج به توضیح زیادی ندارد.

- منوی **Interfaces**

این منو شامل گزینه هایی است که با استفاده از آنها کاربر می تواند رابطهای بین یک گره با گره های دیگر را تعریف کند. در جدول زیر وظیفه هر یک از این گزینه ها به صورت خلاصه فهرست شده اند.

Menu item	Description
Model Attributes	Allows you to edit the node's model-level attributes.
Node Interfaces	Allows you to control various aspects of attribute promotion to the Network level. Only applies to attributes originating in the node's objects, not the node's model attributes.
Node Statistics	Promotes selected module statistics to the node level so that they can be probed without knowledge of node internal structure
Self Description	Allows you to set the information about a node that OPNET uses to match node models to the nodes in an imported topology.

- منوی **Objects**

با استفاده از این منو کاربر می تواند اقدام به انتخاب ماژولهای مختلف برای ایجاد یک مدل جدید کند. در جدول زیر برخی از پر کاربردترین گزینه های این منو لیست شده اند.

[1]- Menu bar

Menu item	Description
Create Processor	Creates a processor module.
Create Queue	Creates a queue module.
Create Packet Generator	Creates a generator module.
Create Clocked Packet Generator	Creates a clocked generator module.
Create Packet Stream	Creates a packet stream.
Create Statistic Wire	Creates a statistic wire.

- منوی Windows

با استفاده از این منو می توان به پنجره های مربوط به سایر ویرایشگرها دست یافت و یا همزمان چندین پنجره را در صفحه به صورت آبشاری مشاهده کرد. در جدول زیر عملکرد گزینه های مختلف این منو را می توان دید.

Menu item	Description
Previous Editor	Makes the previous editor the active window.
Circulate Editors	Moves among open editors.
Hide This Editor	Hides, but does not close or save, the current editor.
Hide Other Editors	Hides, but does not close or save, other editors.
Show All Editors	Shows all previously hidden editors.
System Window	Makes the system window the active window.
Project Editors	Cascading menu lists all open editors of that type. Select the desired model to make the window active.

- منوی Help

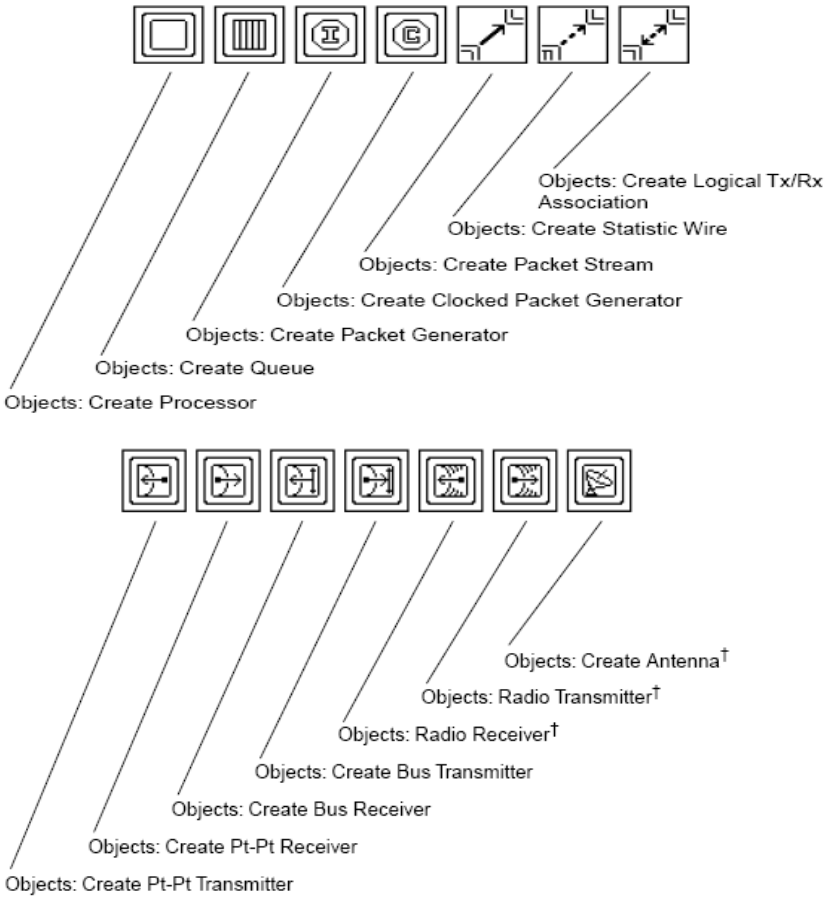
با استفاده از منوی **Help** می توان به فایلهای **pdf** که اطلاعات مفیدی در رابطه با نحوه انجام شبیه سازی در ویرایشگرهای مختلف در اختیار ما می گذارد دست یافت. این فایلها در چند بخش به صورت **online documentation** ، **tutorial** و بخشهای دیگر طبقه بندی شده اند.

در بخش بعدی به صورت خلاصه دکمه های ابزار[1] موجود در محیط ویرایشگر Node را بررسی می کنیم.

[1] - Toll buttons

۳-۲-۳) دکمه های ابزار

برخی از عملکردهای این ویرایشگر که در بخش قبلی توضیح داده شد، در حین ایجاد یک مدل جدید از سوی کاربر به وفور مورد استفاده قرار می گیرند. لذا برای دسترسی سریعتر به این عملکردها، آنها را به صورت دکمه هایی جداگانه در قسمت دکمه های ابزار که در زیر نوار فهرست قرار گرفته است، در اختیار کاربر قرار داده اند. شکل زیر تمامی این دکمه ها را همراه با توضیح مختصری راجع به هر یک نشان می دهد.



۳-۳) ویرایشگر Process

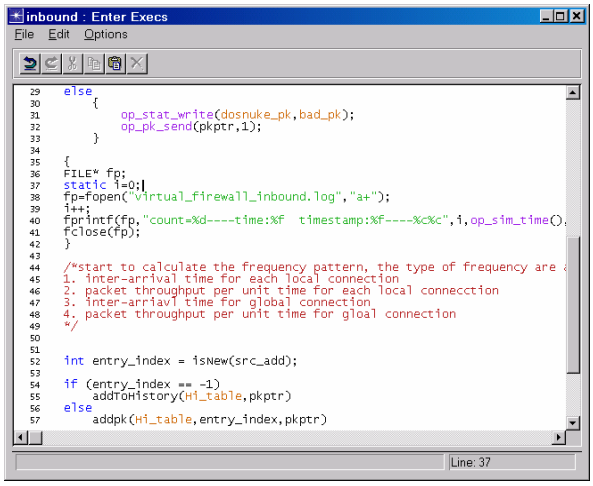
همانطور که قبلا نیز اشاره شد، در ویرایشگر **process** عملکرد هر ماژول تعریف می شود. شبیه سازی مدلها در این ویرایشگر بر مبنای دیاگرامهای **STD** [۱] صورت می گیرد. در نتیجه عملکرد هر ماژول به صورت شماتیک و کاملا قابل درک نشان داده می شود. یک دیاگرام **STD** از تعدادی حالت تشکیل شده است که هر حالت با یک دایره مشخص می شود و با رخداد شرایطی خاص، گذار از حالتی به حالت دیگر صورت می گیرد.

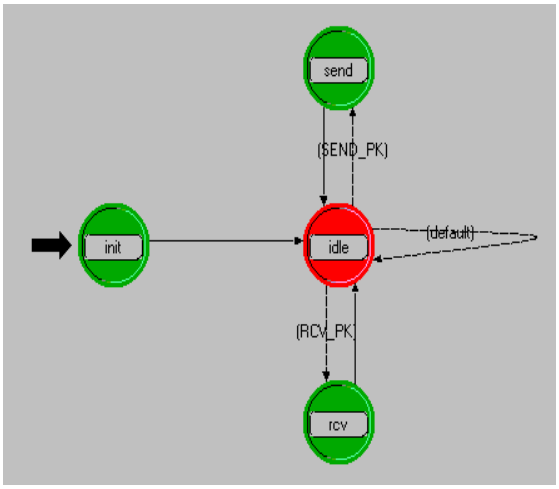
---

†- این دکمه ها فقط در نسخه **OPNET Modeler / Radio** موجوداند.

[۱]- State Transition Diagram

عملکرد هر حالت توسط زبان برنامه نویسی **porto-C** بیان می شود. این زبان برنامه نویسی بر مبنای زبان **C** استوار بوده و با افزودن یکسری توابع خاص به آن که به آنها **Kernel Procedures** گفته می شود و برای شبیه سازی الگوریتمها و پروتکلها طراحی شده است. در شکل زیر نمونه ای از یک مدل





شبیه سازی در این ویرایشگر که شامل دیاگرام **STD** و برنامه **C** متناظر با یکی از حالات آن است را مشاهده می کنید.

۳-۳-۱) حالات اجباری[۱] و غیر اجباری[۲]

حالتهای موجود در دیاگرام **STD** را به دو دسته حالتهای اجباری (دایره های سبز رنگ) و حالتهای غیر اجباری (دایره های قرمز رنگ) تقسیم بندی می کنند. یک حالت اجباری حالتی است که با ورود به آن، پس از اجرای وظایف تعریف شده مربوط به آن، به طور خودکار به حالت دیگری می رویم. به عنوان مثال پس از دریافت یک بسته به یکی از این حالات وارد می شویم، و بعد از آنکه پردازشهای خاصی روی هدر بسته صورت گرفت به حالت دیگری که وظیفه دیگری دارد وارد می شویم.

اما حالتهای غیر اجباری حالتهایی هستند که بعد از ورود به آنها و خاتمه انجام وظایف مربوط به آنها، در همان حالت متوقف می شویم تا اینکه شرایط خاصی رخ دهد. ورود به این حالات معمولا به دلیل انتظار برای وقوع یک وقفه نظیر ورود یک بسته صورت می گیرد.

۳-۳-۲) بررسی منوهای موجود در ویرایشگر **process**

ویرایشگر **process** همانطور که قبلا هم عنوان شد یک محیط مناسب با قابلیتهای فراوان جهت مدلسازی فرایندها و پروتکلهای رایج در شبکه، برای کاربر فراهم می کند. یک راه دسترسی کاربر به این قابلیتها استفاده از منوهای موجود در نوار فهرست است که در بالای صفحه ویرایشگر قرار دارد.

در این بخش به بررسی این منوها می پردازیم:

- منوی **File**:

---

۱ - Forced
۲ - Unforced

این منو شامل عملکردهای سطح بالا نظیر باز کردن و یا بستن یک پروژه شبیه سازی شده، ذخیره تغییرات انجام شده بر روی پروژه و... است.

- منوی **Edit**:

شامل گزینه هایی **Undo**، **Cut**، **Copy**، **Paste** و ... بوده که کاربرد آنها احتیاج به توضیح زیادی ندارد.

- منوی **Interfaces**:

این منو شامل گزینه هایی است که با استفاده از آنها کاربر می تواند رابطهای بین یک مدل با مدلهای دیگر را تعریف کند. در جدول زیر وظیفه هر یک از این گزینه ها به صورت خلاصه فهرست شده اند.

Menu item	Description
Model Attributes	Defines attributes for the process model. These attributes are promoted to processor or queue modules at the node level.
Process Interfaces	Configures certain attributes of processors and queues that rely on the process model at the node level.
Local Statistics	Declares named statistics that are scoped to the queue or processor module and that the process computes and generates.
Global Statistics	Declares named statistics that are scoped to the simulation as a whole and are computed and generated by this process model. Global statistics usually represent system-wide metrics.
Simulation Attributes	Defines attributes that have a simulation-wide scope. These are the simulation attributes that the process model expects to query during the simulation.

- منوی **FSM**:

شامل گزینه هایی برای ایجاد یک دیاگرام **STD** است. برخی از این گزینه ها عبارتند از ایجاد حالت، ایجاد خط گذار و تنظیم حالت اولیه دیاگرام.

- منوی **Code Blocks**:

با استفاده از گزینه های موجود در این منو می توان به متغیرها، توابع مورد استفاده و به طور کلی تمامی موارد مربوط به برنامه **C** نوشته شده برای مدل مربوطه دسترسی داشت. به عنوان مثال با استفاده از گزینه **Function Block** موجود در این منو می توان تمامی توابع استفاده شده در برنامه های **C** نوشته شده برای یک مدل را که در یک پنجره بخ صورت جداگانه لیست می شوند، مشاهده کرد.

- منوی **Compile**:

شامل گزینه هایی برای کامپایل کردن برنامه های نوشته شده است. پس از انجام عملیات کامپایل کردن، با استفاده از گزینه **List Code** که در این منو موجود است، می توان خطاهای احتمالی موجود در برنامه های نوشته شده را ملاحظه کرد و پس از اصلاح آنها مجددا اقدام به کامپایل کردن برنامه کرد.

- منوی **Windows** :

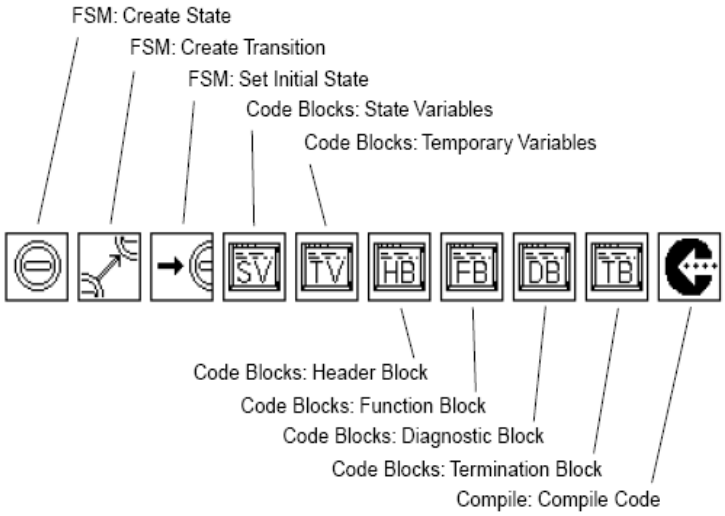
با استفاده از این منو می توان به پنجره های مربوط به سایر ویرایشگرها دست یافت و یا همزمان چندین پنجره را در صفحه به صورت آبشاری مشاهده کرد.

- منوی **Help** :

با استفاده از منوی **Help** می توان به فایلهای **pdf** که اطلاعات مفیدی در رابطه با نحوه انجام شبیه سازی در ویرایشگرهای مختلف در اختیار ما می گذارد دست یافت. این فایلها در چند بخش به صورت **online documentation** ، **tutorial** و بخشهای دیگر طبقه بندی شده اند.

۳-۳-۳) دکمه های ابزار

برخی از عملکردهای این ویرایشگر که در بخش قبلی توضیح داده شد، در حین ایجاد یک مدل جدید از سوی کاربر به وفور مورد استفاده قرار می گیرند. لذا برای دسترسی سریعتر به این عملکردها، آنها را به صورت دکمه هایی جداگانه در قسمت دکمه های ابزار که در زیر نوار فهرست قرار گرفته است، در اختیار کاربر قرار داده اند. شکل زیر تمامی این دکمه ها را همراه با توضیح مختصری راجع به هر یک نشان می دهد.



۳-۳-۴) بررسی توابع تعریف شده در **Porto-C**

همانطور که قبلا هم اشاره شد، زبان برنامه نویسی **Porto-C** که در **OPNET** از آن استفاده می شود، بر مبنای زبان C و با افزودن یکسری توابع و دستورات سطح بالا به آن که به **KPs** معروفند، طراحی شده است. کاربرد آن هم بیشتر در شبیه سازی پروتکلها و الگوریتمها بوده و به همین دلیل محیطی

گرافیکی شامل دیاگرامهای **STD** برای آن در نظر گرفته شده است. در این بخش مروری کوتاه خواهیم داشت بر مهمترین و پرکاربردترین **KPs** موجود در کتابخانه این نرم افزار. در جدول چند نمونه از **KP** های عنوان شده را می توان دید.

Kernel Procedure
op_ima_obj_attr_set()
op_ima_obj_attr_get()
op_pk_fd_set()
op_pk_fd_get()
op_pk_nfd_set()
op_pk_nfd_get()
op_ici_attr_set()
op_ici_attr_get()
op_prg_list_insert()

نامی که برای هر **KP** در نظر گرفته می شود از یک مجموعه اسامی استاندارد انتخاب شده و شامل چند قسمت است. این اسامی به نحوی انتخاب شده اند که بیشترین شباهت را با اسامی رایج در زبان C داشته باشد. ساختار این نامها از چند قانون به صورت زیر پیروی می کند:

نخست آنکه تمامی این اسامی با عبارت "**op_**" شروع می شوند، بدین مفهوم که توسط نرم افزار **OPNET** و برای مدلسازی در این نرم افزار مهیا شده اند. دومین بخش در یک نام **KP** مبین نوع گروهی است که **KP** مورد نظر در آن گروه تقسیم بندی می شود. این گروهها را اصطلاحا **Package** گویند. به عنوان مثال تعداد زیادی از **KP** ها با انجام عملیات بر روی بسته ها سرو کار دارند که تمامی آنها در یک **package** به نام **packet** قرار می گیرند و با حروف اختصاری **pk** نشان داده می شوند. از انواع **package** های دیگر می توان به **intrpt** (که مبین گروه **KP** هایی است که با عملیات وقفه سر وکار دارند) و همچنین **dist** (که مشخص کننده آن گروه از **KP** هایی است که با توزیعهای آماری سر و کار دارند)، اشاره کرد.

بخش سوم نام یک **KP** را می توان به نوعی یک **sub-package** دانست که یک تقسیم بندی اضافه تر درون یک **package** است.

بعد از مقدمه ای که در این خصوص عنوان شد، حال به معرفی مختصرتعدادی از مهمترین **KPs** موجود در **OPNET** می پردازیم. همانطور که در بالا گفته شد، **KP** ها بر مبنای نوع وظیفه ای که بر عهده دارند و عملیاتی که با آن سر و کار دارند، به **package** های مختلفی تقسیم بندی می شوند. ما نیز در اینجا برای معرفی انواع **KPs**، آنها را بر مبنای همین تقسیم بندی معرفی می کنیم.

- گروه اول: **Animation Package**

این گروه از **KP** ها که با حروف اختصاری **Anim** شناخته می شوند، با انجام عملیات خلق تصاویر انیمیشن در حین اجرای شبیه سازی به کار می روند. در زیر تمامی این **KP** ها که در این **package** تقسیم بندی می شوند را می توان دید.

op_anim_igp_circle_fill()
op_anim_igp_drawing_erase()
op_anim_igp_icon_draw()
op_anim_igp_line_draw()
op_anim_igp_macro_draw()
op_anim_igp_macro_redraw()
op_anim_igp_rect_draw()
op_anim_igp_rect_fill()
op_anim_igp_text_draw()
op_anim_ime_gen_pos()
op_anim_ime_nmod_draw()
op_anim_ime_nobj_update()
op_anim_ime_npath_traverse()
op_anim_lprobe_anvid()
op_anim_macro_close()
op_anim_mgp_circle_draw()
op_anim_mgp_circle_fill()
op_anim_mgp_icon_draw()
op_anim_mgp_line_draw()
op_anim_mgp_rect_draw()
op_anim_mgp_rect_fill()
op_anim_mgp_reg_set()
op_anim_mgp_setup_end()
op_anim_mgp_text_draw()
op_anim_mme_nobj_pos()
op_anim_viewer_close()
op_anim_viewer_open()
op_anim_viewer_open_std()
op_anim_viewer_redraw()
op_anim_viewer_scroll()

- گروه دوم: **Distribution Package**

این گروه از **KP** ها که با حروف اختصاری **Dist** شناخته می شوند، با تولید و ایجاد مقادیر تصادفی (آماری) بر مبنای یک توزیع احتمال مشخص مانند توزیع یکنواخت یا توزیع نمائی سر وکار دارند. کاربرد چنین مقادیر تصادفی را می توان در تولید پارامترهای تصادفی مفید در شبیه سازی، نظیر محاسبه زمان ورود بسته ها و وقوع وقفه، تولید آدرس های مقصد بسته ها به صورت تصادفی و غیره دانست. در جدول زیر انواع مختلف این **KP** ها را می توان ملاحظه کرد.

op_dist_exponential()
op_dist_load()
op_dist_outcome()
op_dist_outcome_ext()
op_dist_qwait_est()
op_dist_qwait_estimate()
op_dist_qwait_gg1_handle_create()
op_dist_qwait_mg1_handle_create()
op_dist_uniform()
op_dist_unload()

- گروه سوم: **Event Package**

این گروه از **KP** ها با رویدادهای اتفاق افتاده در طول یک شبیه سازی سرو کار دارند. به عنوان مثال با استفاده از این گروه **KP** ها می توان به حذف و لغو رویدادهایی که وجود آنها بیش از این مورد نیاز نیست اقدام کرد.مثلا در **TCP** هنگامی که منتظر دریافت پیام **Ack** از سوی گیرنده هستیم، رویداد **time-out** را هم مدنظر قرار می دهیم تا اگر بعد از مدت زمان مشخصی این پیغام نرسید، **time-out** رخ دهد. اما اگر پیام **Ack** قبل از موعد مقرر رسید، یا اینکه وقفه دیگری دریافت شد، باید فرایند **time-out** لغو شود. که این کار می تواند توسط **op_ev_cancel()** صورت گیرد. در شکل زیر تعدادی از KP های مربوط به این گروه دیده می شوند.

op_ev_cancel()
op_ev_code()
op_ev_count()
op_ev_count_local()
op_ev_current()
op_ev_dst_id()
op_ev_equal()
op_ev_forced()
op_ev_ici()
op_ev_next()
op_ev_next_local()
op_ev_pending()
op_ev_seek_time()
op_ev_src_id()
op_ev_stat()
op_ev_strm()
op_ev_time()
op_ev_type()
op_ev_valid()

گروه چهارم: Identification Package

این گروه از **KP** ها برای دریافت شماره**ID** هر شیء در شبیه سازی ایجاد شده اند. این **KP** ها از اهمیت ویژه ای برخوردارند. چرا که سایر **KP** ها بر روی اشیائی عمل می کنند که شماره آنها توسط این گروه از **KP** ها تعیین می شود. یک شماره **ID** یک عدد صحیح بوده که هویت یک شیء را در حین اجرای عملیات شبیه سازی به کار می رود. به عنوان مثال **op_id_from_name** سه متغیر را به عنوان آرگومان ورودی خود دریافت نموده و وظیفه تبدیل نام شیء مربوطه به شماره آن را بر عهده دارد. در جدول زیر **KP** های مختلف مربوط به این گروه را مشاهده می کنید.

op_id_from_name()
op_id_from_sysid()
op_id_from_userid()
op_id_self()
op_id_to_type()

گروه پنجم: Interrupt Package

ثبت و تعیین اطلاعات مربوط به وقوع وقفه های ورودی و همچنین تنظیم و زمان بندی وقوع وقفه های خروجی را می توان از جمله مهمترین وظایف این دسته از **KP** ها دانست. به عنوان مثال تابع **op_intrpt_code** کد متناظر با وقفه جاری و فرایند اجرا شده پس از اعمال وقفه را به دست می دهد. این گروه از **KP** ها نیز بسیار پرکاربرد بوده و از اهمیت ویژه ای برخوردارند. در زیر چند دسته از مهمترین این **KP** ها را مشاهده می کنید.

op_intrpt_clear_self()
op_intrpt_code()
op_intrpt_disable()
op_intrpt_enable()
op_intrpt_enable_all()
op_intrpt_force_remote()

گروه ششم: Queue Package

این دسته از **KP** ها با ماژولهای صف و به طور کلی با عملیات مربوط به صف بندی سر و کار دارند. به عنوان مثال **op_q _empty** برای پی بردن به این نکته که صف خالی است یا خیر، به کار می رود. و یا **op_q _ flush** برای خالی کردن یک صف به کار می رود. کاربرد این **KP** در مواردی است که تحت شرایط خاصی بخواهیم بسته های موجود در یک صف که احیانا دچار خطا شده اند و دیگر ارزشی ندارند را حذف کنیم.
در اینجا این گروه از **KP** ها را به صورت لیست شده می توان دید.

op_q_empty()
op_q_flush()
op_q_insert_time()
op_q_stat()
op_q_wait_time()

گروه هفتم: Radio Package

این دسته از **KP** ها با ماژولهای فرستنده و گیرنده در ارتباط هستند و با استفاده از آنها می توان به صورت دینامیک کانالهای مورد استفاده توسط یک فرستنده را تغییر داد. به عنوان مثال در شبیه سازی شبکه های بی سیم، زمانی که یک گره از برد رادیوئی گره فرستنده خارج می شود، باید به نوعی کانالی که پیش از این برای ارسال اطلاعات به آن گره مورد استفاده قرار می گرفت آزاد شده و یا به ارتباط دیگری تخصیص یابد. در شکل زیر انواع مختلف این **KP** ها آورده شده اند. همانطور که می بینیم برخی از آنها برای اضافه کردن (**add**) کانال و برخی دیگر برای حذف کانال (**remove**) به کار می روند.

op_radio_txch_rxch_add()

op_radio_txch_rxch_remove()

op_radio_txch_rxgroup_compute()

op_radio_txch_rxgroup_get()

op_radio_txch_rxgroup_set()

۴- بررسی مدل **TCP** موجود در **OPNET**

همانطور که می دانیم پروتکل **TCP** از جمله مهمترین و پر کاربردترین پروتکلهای لایه انتقال است و بیشتر کاربردهای اینترنت مبتنی بر این پروتکل هستند. این پروتکل که از نوع اتصال گرا[1] بوده، یک مکانیزم مطمئن انتقال بسته ها بر روی یک بستر مخابراتی غیر قابل اطمینان فراهم می کند. اگر به طور خلاصه بخواهیم ویژگیهای این پروتکل را بررسی کنیم، باید موارد زیر را یاد آور شویم:

- قابلیت اطمینان بالای **TCP** در تحویل بسته ها

در **TCP** به دلیل استفاده از مکانیزم تصدیق داده های دریافتی و همچنین به کار بردن مکانیزم ارسال مجدد[2]، انتقال اطلاعات به صورت کاملا اطمینان بخش صورت می گیرد. البته استفاده از این روشها محدودیت عدم امکان استفاده از **TCP** برای کاربردهای مالتی مدیا که ترافیک آنها به زمان حساس است را ایجاد می کند. پس به طور کلی استفاده از **TCP** تنها برای آن دسته از کاربردهایی مناسب است که به تاخیر حساس نبوده و در عوض به دقت بالایی نیاز دارند.

- استفاده از مکانیزمهای کنترل جریان[3]

در **TCP** برای امکان دریافت صحیح بسته ها توسط گیرنده از مکانیزمهای خاصی جهت انجام عملیات کنترل جریان استفاده می شود. کاربرد این امر در مواردی است که به دلیل پر شدن بافر گیرنده و یا کند بودن گیرنده در دریافت اطلاعات نسبت به فرستنده، اطلاعات در بافر گیرنده سرریز شده و از بین می رود. در اینصورت با ارسال مجدد بسته ها از سوی فرستنده، هدر رفتن منابع شبکه و همچنین ایجاد تاخیر بیشتر در دریافت کامل اطلاعات را خواهیم داشت. در **TCP** گیرنده میزان توانایی خود در دریافت اطلاعات را در قالب پیامهای تصدیقی که ارسال می کند، به اطلاع فرستنده می رساند. این کار با استفاده از فیلد **window size** در هدر **TCP** انجام می شود. در این مکانیزم فرستنده به هیچ وجه مجاز به ارسال اطلاعات بیش از مقداری که گیرنده به آن اعلام کرده است نیست.

- استفاده از روش شماره گذاری بسته ها

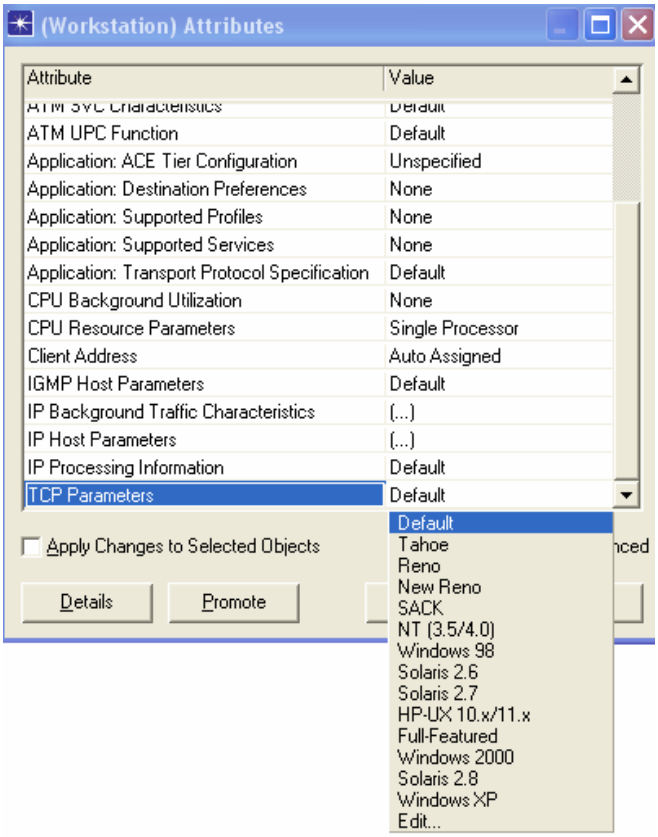
در **TCP** برای کشف بسته های گم شده و همچنین بسته های تکراری، واحد فرستنده برای ارسال بسته ها به آنها شماره ترتیب نسبت می دهد تا گیرنده از روی شماره هر بسته اقدام به کشف بسته های گم شده یا تکراری بپردازد.

---

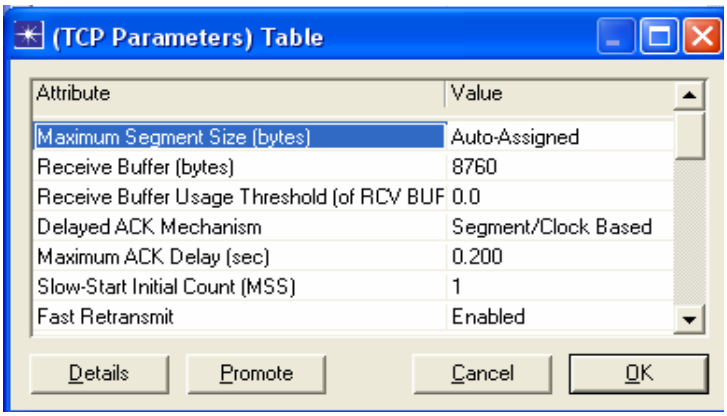
[1] - Connection-Oriented
[2] - Packet Retransmission
[3] - Flow Control

۴-۱) تنظیم پارامترهای TCP در OPNET

در **OPNET** برای هر گره که ساختار لایه ای آن شامل لایه انتقال بوده و به نوعی پروتکل **TCP** را پشتیبانی می کند، یک مجموعه پارامترها تعریف می شود که برای پیکربندی لایه انتقال آن گره، باید این پارامترها به صورت دلخواه تنظیم شوند. در غیر اینصورت مقادیر به صورت پیش فرض از سوی نرم افزار انتخاب می شوند. برای تنظیم این پارامترها می توان بر روی گره مورد نظر در ویرایشگر **Project** کلیک راست کرده و گزینه **Attributes** را انتخاب کرد. در اینصورت پنجره‌ای مشابه شکل زیر باز شده که می توان از بین گزینه های موجود در آن، گزینه **TCP Parameter** را انتخاب کرد. این گزینه خود شامل موارد مختلفی است و برای تعیین دقیق پارامترهای **TCP** باید گزینه آخر (**Edit**) را انتخاب کرد.



با انتخاب گزینه **Edit** پنجره کوچکی مشابه شکل زیر باز شده که در آن پارامترهای مختلف قابل تغییر وجود دارد.



از جمله مهمترین این پارامترها بیشترین طول هر قطعه **TCP** است که می توان آن را به طور دلخواه و بر مبنای پروتکل زیر لایه **MAC** مورد نظر تعیین کرد. همچنین دیگر پارامتر مهم قابل تنظیم، پارامتر بیشترین تاخیر در ارسال پیام تصدیق وصول داده ها است. توجه داشته باشید که در **TCP** به محض دریافت یک بسته، پیام تصدیق آن فرستاده نمی شود. بلکه به امید اینکه تا لحظاتی بعد داده ای برای ارسال ظاهر شود و همراه با آن، پیام تصدیق داده های دریافتی قبلی نیز ارسال شود. به بیان دیگر پهنای باند را بی جهت و با ارسال پیامهای فاقد بخش داده هدر ندهیم.

از دیگر ویژگیهای قابل تنظیم در این منو، سطح آستانه بافر گیرنده است. با تغییر این پارامتر، طول پنجره پیشنهادی (پنجره ای که برای کنترل جریان در **TCP** استفاده می شود) تغییر خواهد کرد. به نحوی که با بزرگ کردن آن می توان حجم بافر گیرنده را افزایش داده و مسئله کنترل جریان را کمرنگ کرد. و برعکس با کوچک کردن این پارامتر حجم بافر گیرنده کاهش پیدا کرده و مسئله کنترل جریان از اهمیت ویژه ای برخوردار خواهد گشت.

همانطور که می دانیم در **TCP** از الگوریتم شروع آهسته[۱] برای ارسال اطلاعات استفاده می شود. یکی دیگر از موارد قابل تنظیم در این منو، مقدار اولیه پنجره ازدحام[۲] در هنگام شروع به کار الگوریتم شروع آهسته است. در بیشتر پیاده سازیهای **TCP** این مقدار برابر یک **MSS**[۳] در نظر گرفته می شود.

مورد دیگری که در این پنجره مشاهده می شود و می توان آن را به دلخواه تعیین کرد، الگوریتمها و پروتکلهای جانبی است که ممکن است در بعضی از نسخه های **TCP** به کار روند. از جمله معروفترین این الگوریتم ها، می توان به الگوریتم ناگل، الگوریتم ارسال مجدد سریع و الگوریتم کارن اشاره کرد. در ضمن برای مشاهده جزئیات مربوط به هر الگوریتم، می توان با انتخاب گزینه **Details** در قسمت پایین پنجره، جزئیات بیشتری در مورد هر یک از آنها به دست آورد.

---

۱ - Slow Start
۲ - Congestion Window
۳ - Maximum Segment Size

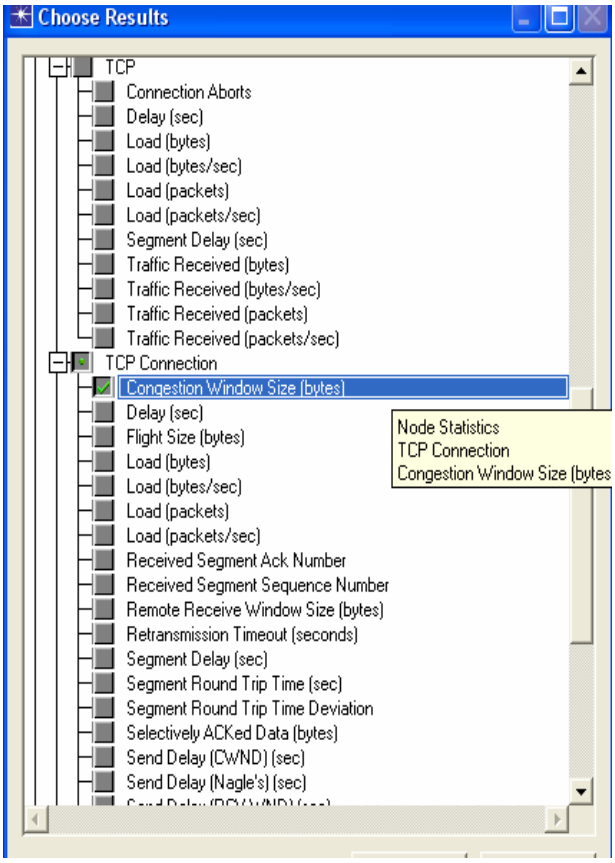
۲-۴) انتخاب پارامترهای موردنظر برای شبیه سازی در TCP

برای بررسی و آنالیز عملکرد یک اتصال **TCP** ، پارامترهای زیادی وجود دارند که می توان با بررسی آنها در حین اجرای شبیه سازی، عملکرد یک اتصال **TCP** را مورد ارزیابی قرار داد. این پارامترها را در یک تقسیم بندی کلی، می توان در گروه مختلف بررسی کرد:

نخست آن دسته از پارامترهایی که فقط مربوط به یک اتصال **TCP** در یک گره بوده مانند طول پنجره ازدحام مربوط به یک اتصال خاص.

اما دسته دوم این پارامترها عمومی بوده و مربوط به تمامی اتصالات TCP است که ممکن است یک گره با سایر گره ها در یک شبکه داشته باشد. بررسی این دسته از پارامترها قابلیت تحلیل کل ترافیک منتقل شده از لایه انتقال یک گره در شبکه را به ما می دهد.

اما برای تعیین یکی از این پارامترها جهت شبیه سازی و جمع آوری آمار از آن در حین شبیه سازی، می توان در ویرایشگر **project** با انتخاب گزینه **choose statistics** از منوی **simulation** پنجره ای که مربوط به این کار است را باز کرد. این پنجره در شکل زیر نشان داده شده است.

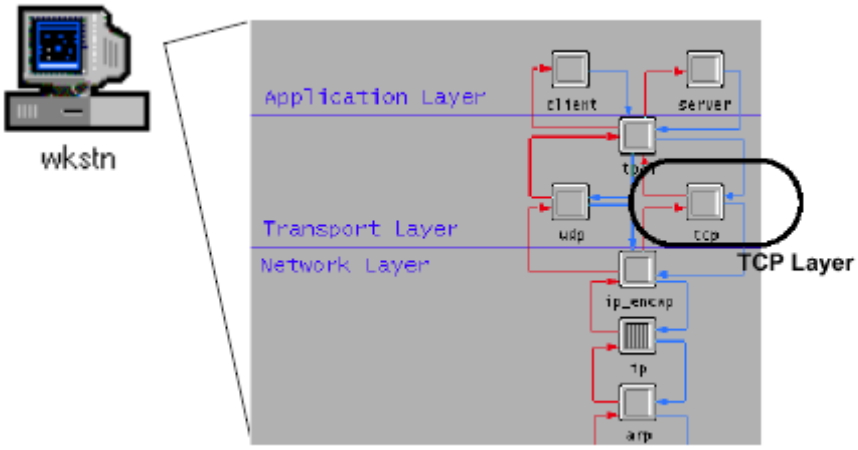


حال می توان در لیست موجود در این پنجره، پارامترهای مربوط به پروتکل **TCP** را انتخاب کرد. به عنوان مثال در شکل فوق پارامتر اندازه پنجره ازدحام انتخاب شده است. پس از اجرای شبیه سازی، می

توان با انتخاب گزینه **View Result** از منوی **Result** نتایج حاصله و آمارهای اخذ شده از پارامتر مورد نظر در حین اجرای شبیه سازی را مشاهده کرد.

۳-۴) ساختار مدل TCP در OPNET

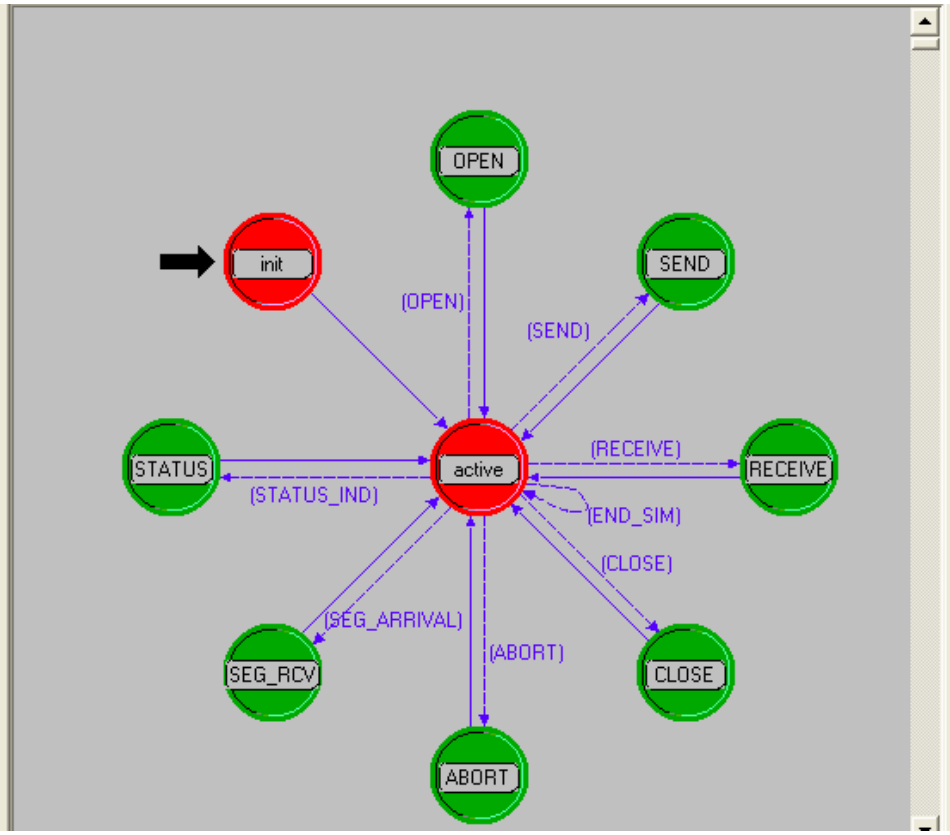
در این فصل به بررسی دقیقتر جزئیات مدل **TCP** موجود در **OPNET** می پردازیم. به طور کلی هر گره در شبکه که ساختار لایه ای آن شامل لایه انتقال بوده، پروتکل **TCP** در ساختار آن اجرا شده است. ساده ترین این ادوات را می توان یک کامپیوتر عادی دانست که می تواند به شبکه متصل شود و در شکل زیر شمای آن را در ویرایشگر **Project** می بینیم. به طور کلی چنین عنصری در شبکه را یک ایستگاه کاری[۱] گویند. اگر در محیط ویرایشگر **project** بر روی یک این عنصر کلیک کنیم، ساختار لایه ای آن را در محیط ویرایشگر **Node** مطابق شکل زیر خواهیم دید. همانطور که در شکل مشخص است، ماژول مربوط به پروتکل **TCP** در ناحیه مربوط به لایه انتقال قرار دارد. البته به موازات آن ماژولی برای پروتکل **UDP** قرار گرفته که این عنصر را قادر می سازد تا یک پروتکل مناسب برای کاربردهای بلادرنگ نظیر کاربردهای مالتی مدیا در لایه انتقال خود داشته باشد.



همچنین با کلیک کردن بر روی ماژول **TCP** در ویرایشگر **Node** ، می توان ساختار دقیق این پروتکل را در قالب یک دیاگرام حالت در ویرایشگر **process** مشاهده کرد. این دیاگرام را در شکل زیر مشاهده می کنید. این دیاگرام از دو حالت غیر اجباری (قرمز رنگ) و هفت حالت اجباری تشکیل شده است. البته نمای ظاهری این دیاگرام با آنچه در **RFC** مربوط به **TCP** آمده است اندکی متفاوت است، ولی در عمل دو دیاگرام با هم تفاوتی ندارند.

---

[۱] - Workstation



در شکل زیر دیاگرام حالت متناظر با پروتکل **TCP** که از سوی ارائه کنندگان آن در **RFC** شماره ۷۹۳ آمده است را ملاحظه می کنید. این دیاگرام با فرض یک ارتباط مشتری- سرویس دهنده رسم شده است. خطوط توپر و پررنگ مبین گذار حالات مشتری و خطوط نقطه چین مبین گذار حالات سرویس دهنده است.

ابتدا به بررسی اتفاقات رخ داده در سمت مشتری می پردازیم. در ابتدای کار، در حالت **CLOSED** هستیم. به مفهوم آنکه هنوز هیچ اتصالی بین مشتری و سرویس دهنده برقرار نشده است. آنقدر در این حالت می مانیم تا تابع **CONNECT** در سمت مشتری اجرا شود. این بدان مفهوم است که از لایه بالاتر پیامی دال بر برقراری یک اتصال با سرویس دهنده رسیده است. پس از اجرای این تابع، یک

قطعه **TCP** که در هدر آن و در فیلد **SYN** عدد یک قرار گرفته است برای سرویس دهنده ارسال می شود. لازم به ذکر است که ارسال پیامی با **SYN=**۱ به مفهوم تقاضای ایجاد یک اتصال است. پس از ارسال این پیام برای سرویس دهنده، به حالت **SYN SENT** می رویم. در این حالت مشتری منتظر می ماند تا از سوی سرویس دهنده پیامی مبتنی بر موافقت آن با برقراری اتصال می ماند. در اینجا برای برقراری اتصال و گذار به حالت **ESTABLISHED** مشتری باید یک پیام **Ack** با **SYN=**۱ از سرویس دهنده دریافت کند. پس از دریافت این پیام، مشتری نیز یک پیام **Ack** دال بر تصدیق دریافت پیغام سرویس دهنده برای آن ارسال می کند و به حالت **ESTABLISHED** می رود. در این حالت اتصال به صورت دوطرفه برقرار شده و تبادل اطلاعات می تواند بر مبنای جزئیات پروتکل **TCP** صورت پذیرد. در واقع مراحل عنوان شده در برقراری یک اتصال را می توان مراحل دست تکانی[۱] ۳ مرحله ای دانست.

با اجرای تابع **CLOSED** در سمت مشتری (به مفهوم تقاضا برای خاتمه اتصال) یک قطعه **TCP** با **FIN=**۱ برای سرویس دهنده ارسال می شود و دیاگرام به حالت **FIN WAIT** ۱ می رود. مجددا لازم به یادآوری است که ارسال پیامی با **FIN** =۱ به مفهوم تقاضای خاتمه اتصال است. در حالت **FIN WAIT** ۱ مشتری منتظر دریافت پیام **Ack** از سوی سرویس دهنده است تا ارتباط از یک طرف رسما قطع شود. در این حالت مشتری به حالت **FIN WAIT** ۲ می رود. در این حالت همانطور که عنوان شد، تبادل اطلاعات فقط از سمت سرویس دهنده به مشتری صورت می گیرد.

در نهایت دیاگرام آنقدر در این حالت می ماند تا سرویس دهنده نیز اقدام به خاتمه اتصال کند. این کار با دریافت یک پیام با **FIN** =۱ توسط مشتری از سوی سرویس دهنده صورت می گیرد. مشتری نیز با دریافت این پیام، تصدیق آن را برای سرویس دهنده ارسال کرده و به حالت **TIME WAIT** می رود. در این حالت پس از گذشت زمان محدودی اتصال از دو طرف به صورت کامل قطع می شود. حالت **TIME WAIT** را می توان حالت انتظار برای از بین رفتن تمام بسته های سرگردان دانست.

حال به بررسی اتفاقات رخ داده در سمت سرویس دهنده می پردازیم. در اینجا نیز ابتدا در حالت **CLOSED** هستیم و با اجرای تابع **LISTEN** به حالت **LISTEN** می رویم. در این حالت سرویس دهنده در حال گوش کردن به پورت مربوطه جهت دریافت یک پیام با **SYN** =۱ است. با دریافت این پیام، سرویس دهنده نیز پیام تصدیقی با **SYN=**۱ برای مشتری فرستاده و به حالت **SYN RCVD** می رود. در این حالت سرویس دهنده متنظر دریافت پیام تصدیق از سوی مشتری است. با دریافت این پیام، مرحله آخر از مراحل دست تکانی سه مرحله ای نیز انجام شده و به حالت **ESTABLISHED** خواهیم رفت. در این مرحله با دریافت یک پیام با **FIN=**۱ از سوی مشتری، پیام تصدیقی برای آن ارسال شده و به حالت **CLOSE WAIT** خواهیم رفت. در این حالت اتصال از سمت مشتری به

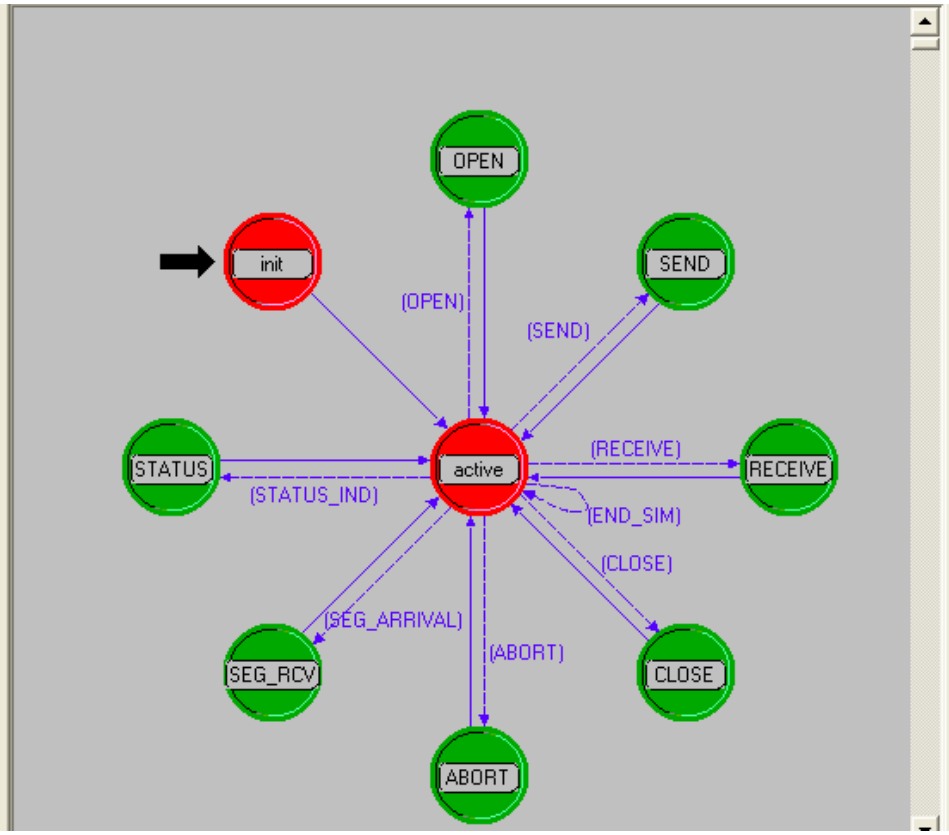
---

[۱] - Hand shaking

سرویس دهنده قطع شده است. در این حالت آنقدر خواهیم ماند تا اینکه تابع **CLOSE** در سمت سرویس دهنده اجرا شود. با اجرای این تابع یک پیام با **FIN** =۱ برای مشتری ارسال شده و به حالت **LAST Ack** خواهیم رفت. در این حالت سرویس دهنده منتظر دریافت **Ack** از سوی مشتری می ماند تا او نیز از سمت خود اتصال را قطع نماید.

در دیاگرام فوق برخی از گذارهای دیگر نیز ممکن است در اثر وقوع شرایط خاصی صورت پذیرند که در شکل با خطوط توپر و نازک نشان داده شده است.

حال که دیاگرام حالت معرفی شده برای پروتکل **TCP** در **RFC** شماره ۷۹۳ را بررسی کردیم، به معرفی این دیاگرام در **OPNET** می پردازیم.



همانطور که در شکل می بینیم این دیاگرام از دو حالت غیر اجباری و هفت حالت اجباری تشکیل شده است. به طور کلی ارتباط بین لایه **TCP** و لایه کاربرد بر مبنای دو نوع دستور صورت می گیرد. اگر ارتباطی بخواهد از لایه کاربرد به لایه **TCP** برقرار شود، به وسیله یک مجموعه دستورات مشخص که آنها را **command** گوییم، این کار انجام می شود. و برعکس اگر **TCP** بخواهد به لایه کاربرد پیامی ارسال کند، با استفاده از یکسری علائم قراردادی که به آنها **indication** گفته می شود، این

کار را انجام می دهد. مجموعه **command** های مورد استفاده توسط لایه کاربرد برای صحبت با **TCP** عبارتند از **OPEN**، **SEND**، **RECEIVE**، **CLOSE** و **ABOART** و مجموعه indication های مورد استفاده در **TCP** برای برقراری ارتباط با لایه کاربرد عبارتند از **SEG_FWD**، **ESTAB**، **CLOSED** و **ABOARTED**.