Rapport Final

Analyseur de Réseau Nmap

Document rédigé par :

Mohamed Guelleh

Et

Mactire Osman

Présenté à :

Kamel Adi Ph.D

Professeur Superviseur

Et

Michal Iglewski

Professeur Coordonnateur

Dans le cadre du cours :

Projet de synthèse

INF4173

Département d'informatique et d'ingénierie

Université du Québec en Outaouais

Vendredi 20 Avril 2007.

Table des matières

But du projet :

Ce projet a était proposé par Dr. Kamel Adi dans le cadre du cours Projet de Synthèse. Ce projet a pour but de développer et d'intégrer une application permettant de tracer une configuration de réseau (systèmes, topologie, etc.) à partir d'information contenus dans un document XML et généré par le logiciel d'analyse de réseaux NMAP.

Objectif principal :

L'objectif principal est de créer une interface qui va permette à un utilisateur de visualiser de manière interactive et de naviguer dans une représentation d'un réseau quelconque. Cette application devra avoir entre autre la possibilité d'obtenir une vue globale du réseau et des vues locales par des fonctions de «zoom».

Problématique :

Qu'est-ce que Nmap?

Nmap qui veut dire « Network Mapper » est un outil de balayage de ports. C'est un logiciel gratuit. Il est distribué par Insecure.Org. Il a été conçu pour scanner rapidement des grands réseaux; mais il fonctionne aussi très bien sur une cible unique.

En premier lieu, il nous fallait bien comprendre les différentes fonctionnalités de ce logiciel. Nmap utilise diverses techniques d'analyse basées sur des protocoles tels que TCP, IP, UDP ou ICMP pour scanner les ports d'un ordinateur distant.

Quel outil de programmation ?

Nous avons cherché un outil qui nous permettrait d'afficher les données du fichier XML en interface graphique. Au début, nous avons pensé utiliser XUL qui est un langage XML prévu pour réaliser un ensemble de balises permettant de définir des boutons, des listes, des menus, ou encore des zones d'édition, bref tous les éléments d'une véritable interface utilisateur.

En effet, les utilisations possibles de XUL consistent à une simple page Web en remplaçant le HTML par XUL et les objets XUL se manipulent via une interface DOM, comme en DHTML. De plus, ces données peuvent être récupérées en XML et de manière asynchrone.

Le plus grand désavantage de XUL qui nous a forcé de délaisser cette technologie est que son interface graphique ne fonctionne qu'avec les navigateurs Web Mozilla et Netscape .Malheureusement la majorité des utilisateurs de navigateur web utilisent le navigateur Explorer ; d'ou notre réticence envers la technologie XUL.

Nous nous sommes pas arrêter juste sur XUL, nous avons cherché d'autres outils qui nous permettraient d'affiche une interface conviviale comme JGraph.

Méthodologie (aspects généraux) :

- Approche globale

- Description de l'application

- processus de développement
    - Étude préalable :
        - Recueil des informations
        - Diagramme des flux
    - Étude détaillée
        - *Modèle logique des données*
    - Réalisation

- Mise en œuvre
  Moyens techniques (locaux, matériel informatique, fournitures, etc.).

Les étapes spécifiques de notre projet

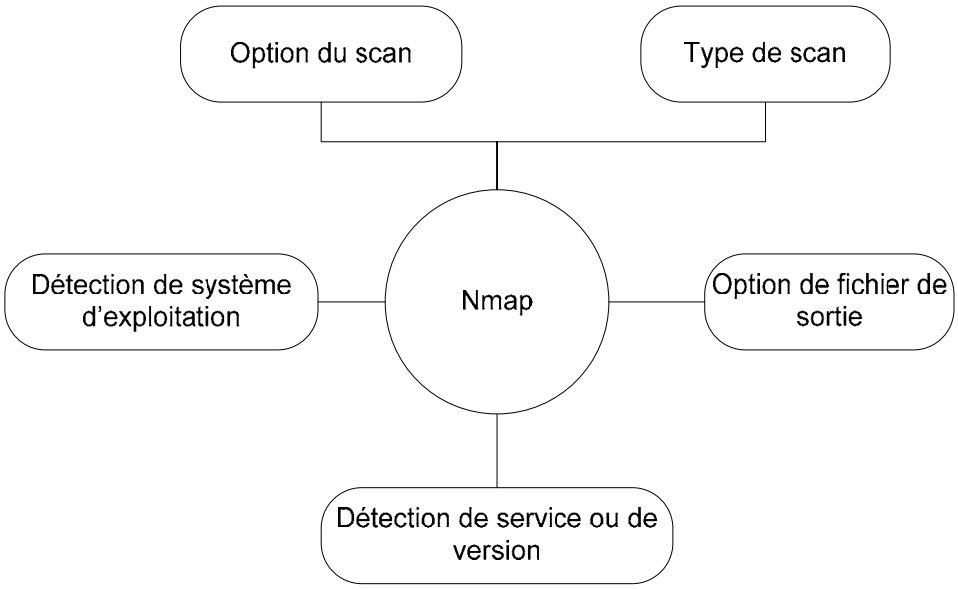
Comprendre et maîtriser le logiciel NMAP

Nmap offre plusieurs techniques de balayage. Dans notre projet, nous avons travailler sur un nombre limité d'entre elles pour mieux maîtriser les aspects du protocole TCP.

L'idée, c'est de pouvoir observer les différents usages de Nmap pour obtenir des informations sur les systèmes et en même temps de découvrir les traces laissées par le balayage de Nmap du côté de la cible.

Voici les différents paramètres que Nmap peut prendre :

Nmap [Types de Balayage ...] [Options] {spécifications des cibles}



Quelques exemples de commande

Voir tous les ports TCP ouverts sur une machine, utilisation de messages SYN, donc pas de log sur la machine cible :

- nmap -sS 127.0.0.1

Voir tous les ports UDP ouverts sur une machine :

- nmap -sU 127.0.0.1

Voir si une machine est sur le réseau (scan Ping) :

- nmap -sP 127.0.0.1

Scanner une plage d'adresses. Ici toutes les adresses de 192.168.0 à 192.168.255 :

- nmap 192.168.0,.0-255

Connaitre le système d'exploitation de la machine (TCP/IP fingerprint) :

- nmap -O 127.0.0.1

Scanner un port précis. Ici, c'est le port http :

- nmap -p 80 127.0.0.1

Scanner une plage de ports. Ici on balaye du port 0 au 80 et tous ceux supérieurs à 60000) :

- nmap -p 0-80,60000 127.0.0.1

Scanner des serveurs web au hasard sur le réseau :

- nmap -v -sS -iR 0 -p 80

Désactiver la résolution DNS inverse des hôtes, augmente la rapidité :

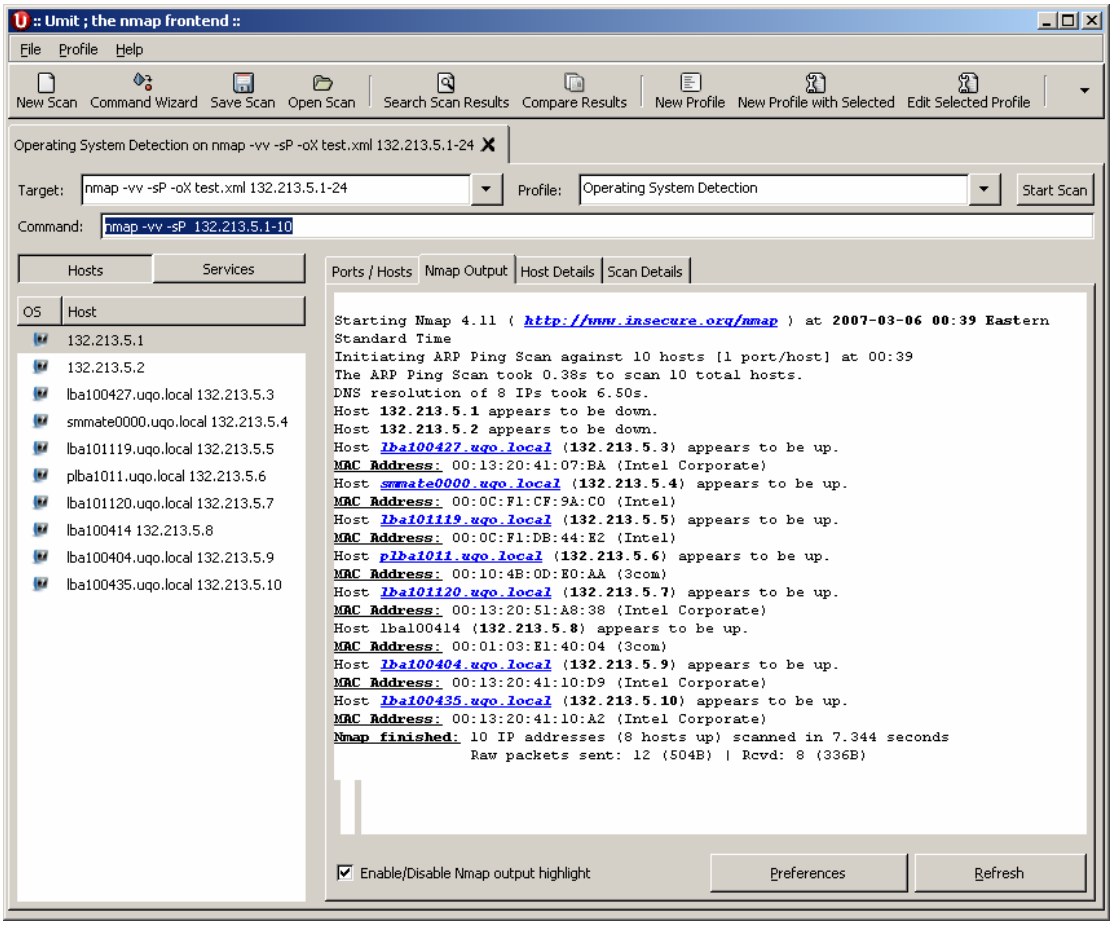
- nmap -n 127.0.0.1

Choisir un fichier de sortie pour y écrire les résultats du balayage :

- nmap -oX resultat.xml 127.0.0.1

Nous allons voir plus en détail deux logiciels qui nous permettent de balayer des stations dans un réseau.

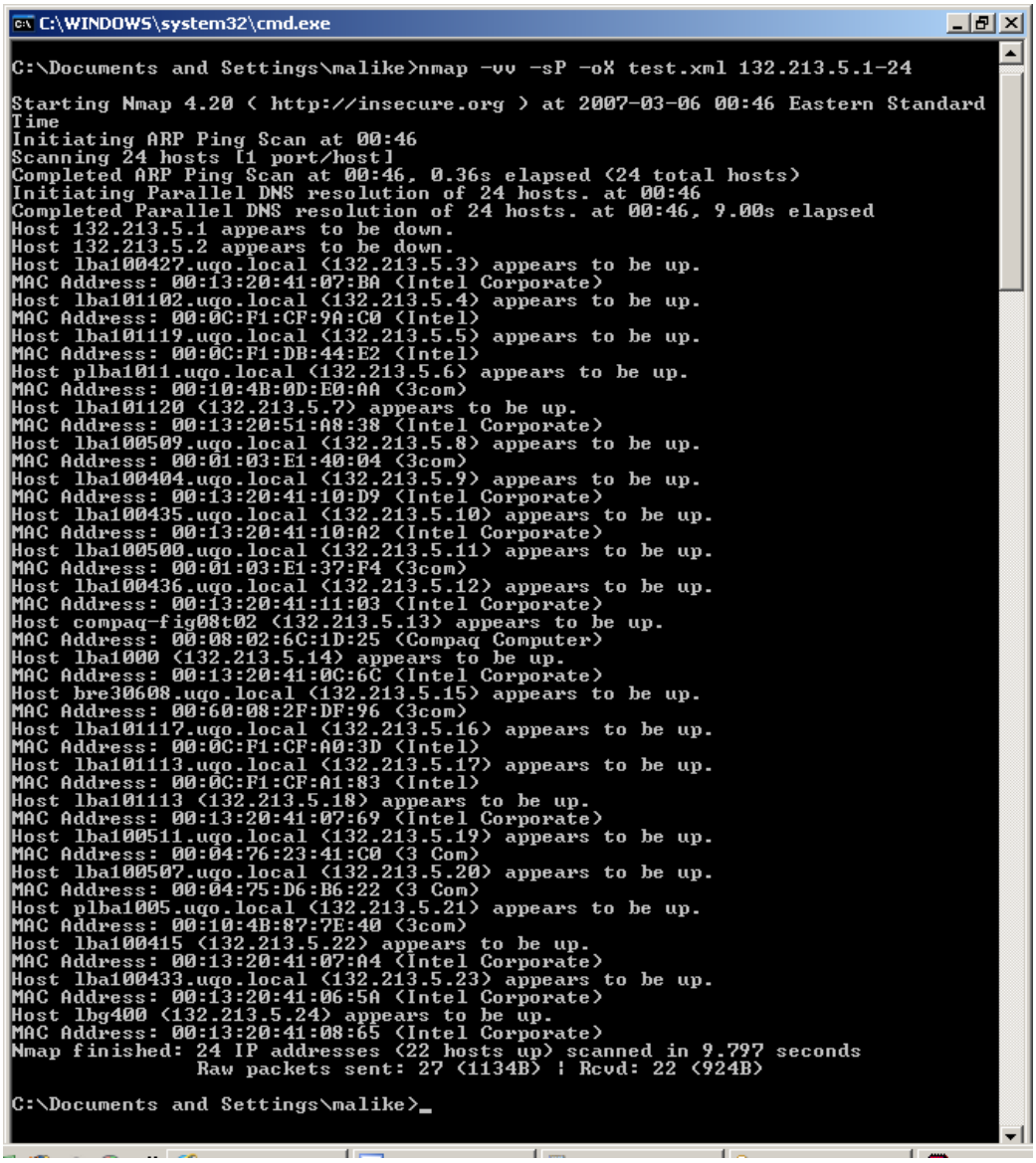
Exemple d'exécution du logiciel UMIT :



UMIT est une nouvelle commande de dialogue graphique de Nmap. Elle est vraiment utile pour les utilisateurs avancés et faciles pour être employé par des internautes débutants. Avec UMIT, un administrateur de réseau pourrait créer des profils de balayage très rapidement.

L'avantage de ce logiciel est qu'il a une interface graphique très conviviale. Mais, sa manipulation est discutable car, il ne permet pas l'option de sauvegarder la sortie du fichier en extension XML.

Exemple d'exécution du logiciel Nmap :



Nmap n'a pas une interface graphique comme UMIT mais par contre il nous permet très facilement de sauvegarder un fichier en XML.

Comprendre le Format de Sortie de Nmap

Nmap nous offre quelques formats de sorties. Le format par défaut est interactif en sortie et il est envoyé en sortie standard (stdout).
On retrouve aussi le format de sortie, qui est semblable au premier format mais il affiche moins d'informations et d'alertes étant donné qu'il est plutôt destiné à être analysé à la fin des balayages au lieu de la manière interactive.

La sortie du format XML est l'une des plus importantes qui peut être converti en HTML. Elle est facilement traitée par des programmes tiers comme les interfaces graphiques pour Nmap, ou importée au sein de bases de données.

Voici un exemple d'exécution de la commande de Nmap pour un fichier de sortie :

nmap -oX myscan.xml Target

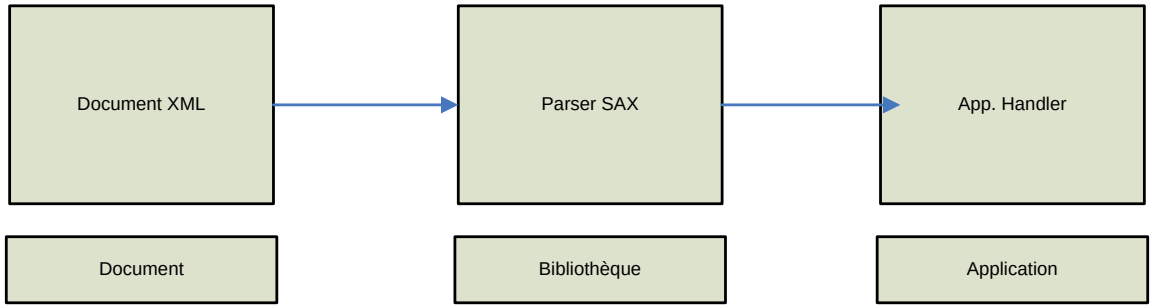
Maîtriser des outils de parsing XML

En effet, aucune API n'est meilleure que les autres, tout est question de besoins. Pour résumer, on peut dire que :

- SAX est conviviale à programmer, rapide et peu gourmand en ressources (UCT et mémoire). Il ne permet pas cependant de manipuler la structure du document.
- DOM est complexe à mettre en oeuvre et gourmand en ressources mais permet la manipulation de la structure du document.
- JDOM est plus facile à mettre en oeuvre que DOM, mais n'est pas encore un standard et est encore en phase de développement.
- JAXP est très utile pour encapsuler l'instanciation d'un parseur DOM, mais on peut s'en passer pour les parseurs SAX.

Dans notre projet nous avons utilisé le parseur Sax car nous n'avions pas besoin de manipuler une structure ni de la garder en mémoire.

Voici le fonctionnement graphique du parseur SAX.



Voici le fichier XML de sortie après l'exécution du logiciel NMAP

```
<?xml version="1.0" ?>
<?xml-stylesheet href="nmap.xsl" type="text/xsl"?>
<!-- Nmap 4.20 scan initiated Sat Feb 24 17:34:31  -->
<nmaprun scanner="nmap" args="nmap -vv -sP -oX test.xml 132.213.5.1-24"
start="1172356471" startstr="Sat Feb 24 17:34:31 2007" version="4.20"
xmloutputversion="1.01">
<verbose level="2" />
<host>
        <status state="down" type="server"/>
        <address addr="132.213.5.1" addrtype="ipv4" />
        <connect  />
</host>

<host>
        <status state="up" type="computer"/>
        <address addr="132.213.5.2" addrtype="ipv4" />
        <connect connectAddr="132.213.5.1" />
        <address addr="00:05:5D:D2:02:6B" addrtype="mac" vendor="D-Link
        Systems" />
        <hostnames />
</host>

<host>
        <status state="up" type="computer"/>
        <address addr="132.213.5.3" addrtype="ipv4" />
        <connect connectAddr="132.213.5.1" />
        <address addr="00:13:20:41:07:BA" addrtype="mac" vendor="Intel
        Corporate" />
        <hostnames />
</host>
```

```
<host>
        <status state="up" type="computer"/>
        <address addr="132.213.5.4" addrtype="ipv4" />
        <connect connectAddr="132.213.5.1" />
        <address addr="00:0C:F1:CF:9A:C0" addrtype="mac" vendor="Intel" />
        <hostnames><hostname name="lba101102.uqo.local" type="PTR"/>
        </hostnames>
</host>

<host>
        <status state="up" type="computer"/>
        <address addr="132.213.5.5" addrtype="ipv4" />
        <connect connectAddr="132.213.5.1" />
        <address addr="00:13:20:41:07:69" addrtype="mac" vendor="Intel
        Corporate" />
        <hostnames><hostname name="lba100402.uqo.local" type="PTR"
        /></hostnames>
</host>
…

…

…

<runstats><finished time="1172356521" timestr="Sat Feb 24 17:35:21
2007"/><hosts up="23" down="1" total="24" />
<!-- Nmap run completed at Sat Feb 24 17:35:21 2007; 24 IP addresses (23
hosts up) scanned in 50.125 seconds -->
</runstats></nmaprun>
```

Programmation des fonctionnalités de Parser SAX

Voici comment fonctionne Parser SAX dans notre application.

Notre défi était de récupérer un fichier XML quelque soit sa taille et de le convertir en interface graphique.

```
////////////////////////////////////////////////////////
// Les gestionnaires d'évènements SAX //////////////////////
////////////////////////////////////////////////////////

    public void startDocument () throws SAXException {
      System.out.println("START DOCUMENT");
      stParent = null;
      tabCellParent = new Vector <DefaultGraphCell>();
```

```
  parentRoot = null;
 }

public void endDocument () throws SAXException {

  System.out.println("END DOCUMENT");

}

public void startElement (String namespaceURI,
                    String sName, // simple name
                    String qName, // qualified name
                    Attributes attrs)
                    throws SAXException  {


  if (qName == "host" ){
        addr1 = null;     type1 = null;     typeDeNoeud = null;
        addr2 = null;     type2 = null;     connectAddr = null;

  }


  if (qName == "address" ){
        if (attrs != null ){
              for (int i = 0; i < attrs.getLength (); i++) {

                    if (attrs.getQName(i)== "addr"){
                          if(addr1 == null)
                                addr1 = attrs.getValue (i);
                          else
                                addr2 = attrs.getValue (i);
                    }

                    if (attrs.getQName(i)== "addrtype"){
                          if(type1 == null)
                                type1 = attrs.getValue (i);
                          else
                                type2 = attrs.getValue (i);
                    }
              }// fin du for
          }
    }// fin du if adresse



  if (qName == "connect" ){
        if (attrs != null )
              for (int i = 0; i < attrs.getLength (); i++) {
                    connectAddr = attrs.getValue(i);
              }
  } // fin connect

  if (qName == "status" ){
        if (attrs != null )
              for (int i = 0; i < attrs.getLength (); i++) {
                    if (attrs.getQName(i)== "type")
```

```
                                typeDeNoeud = attrs.getValue(i);
                        }
          } // fin status

        }// fin startElement

        public void endElement (String namespaceURI,
                                String sName, // simple name
                                String qName ) // qualified name
                                throws SAXException {

          if (qName == "host" ){
                DefaultGraphCell cell = new DefaultGraphCell();
                DefaultPort  portSource = new DefaultPort();
                DefaultPort  portTarget = new DefaultPort();
                DefaultEdge lien = new DefaultEdge();
                Map donnee = new Hashtable();

                String ip = "ip";
                String ipconnect = "ipconnect";
                String addMac = "addMac";
                String typeConnect = "typeConnect";

                      donnee.put((Object) ip ,(Object) addr1 );

                      if(connectAddr != null)
                            donnee.put((Object) ipconnect ,(Object)
                      connectAddr );

                      if(addr2 != null)
                            donnee.put((Object) addMac ,(Object) addr2 );

                      donnee.put((Object) typeConnect ,(Object) typeDeNoeud
);


                      cell.setAttributes(new AttributeMap( donnee));
                      if(addr1 != null){


                      myAttribute.put(cell, cell.getAttributes());
                      GraphConstants.setChildrenSelectable
                      (cell.getAttributes(), true);
                      GraphConstants.setBounds(cell.getAttributes(),new
                      Rectangle(640, 20,50, 50));
                      GraphConstants.setIcon(cell.getAttributes(),
                      image(typeDeNoeud));
                      GraphConstants.setBackground(cell.getAttributes(),
                      Color.blue);
                      GraphConstants.setOpaque(cell.getAttributes(), true);
                      GraphConstants.setBorder(cell.getAttributes(),
                      BorderFactory.createRaisedBevelBorder());
                      //GraphConstants.setBorderColor(cell.getAttributes(),
                      null);


                }
```

```
//    if(traceParent == null){cellParent =
      cell;traceParent=addr1;}

      if(connectAddr != null){

            //if(stParent != connectAddr)

      //    String ip = "lienParent";
      //    donnee.put((Object) ip ,(Object)connectAddr );
      //    cell.setAttributes(new AttributeMap( donnee));

                  //cellParent = cell;
      }
      cell.add(portTarget);
      lien = setEdge(GraphConstants.ARROW_CLASSIC,true);
      //DefaultGraphCell parent = getParent(cell);
      if(parentRoot != null){

            if(connectAddr.compareTo((String)parentRoot.getAttribu
      tes().get("ip"))!= 0){
                  if(!tabCellParent.isEmpty()){
                        int i;
                        for(i=0 ; i < tabCellParent.size(); i++){
                        java.util.List children =
                        tabCellParent.get(i).getChildren();

                      if (children != null) {
                          Iterator iter = children.iterator();

                          while (iter.hasNext()) {
                              Object obj = iter.next();

                              if (obj != null && obj  instanceof
                              DefaultPort) {

                       if( connectAddr.compareTo(
                      (String)((DefaultGraphCell)obj).getAttributes
                      ().get("ip") ) == 0  ){
                         parentRoot = (DefaultGraphCell)obj;
                         tabCellParent.add((DefaultGraphCell)obj);
                  //return; // sortie de la boucle for
                  }

                                    }
                              }// fin du while
                      }
                        }// fin du for
                        if(i == tabCellParent.size()){
                              //tabCellParent.add(cell);
                              //parentRoot = cell;
                        }
                  }
            }
            parentRoot.add(portSource);
            parentRoot.setAllowsChildren(true);
```

```
                insertModel(lien, portSource, portTarget, parentRoot
                ,cell, null, null);
                }else{
                tabCellParent.add(cell);
                parentRoot = cell;
          }
          //Jgraph.getGraphLayoutCache().insert(cell);

    }// fin du host

}// fin de endElement
```

Intégrer ces outils dans une application Java

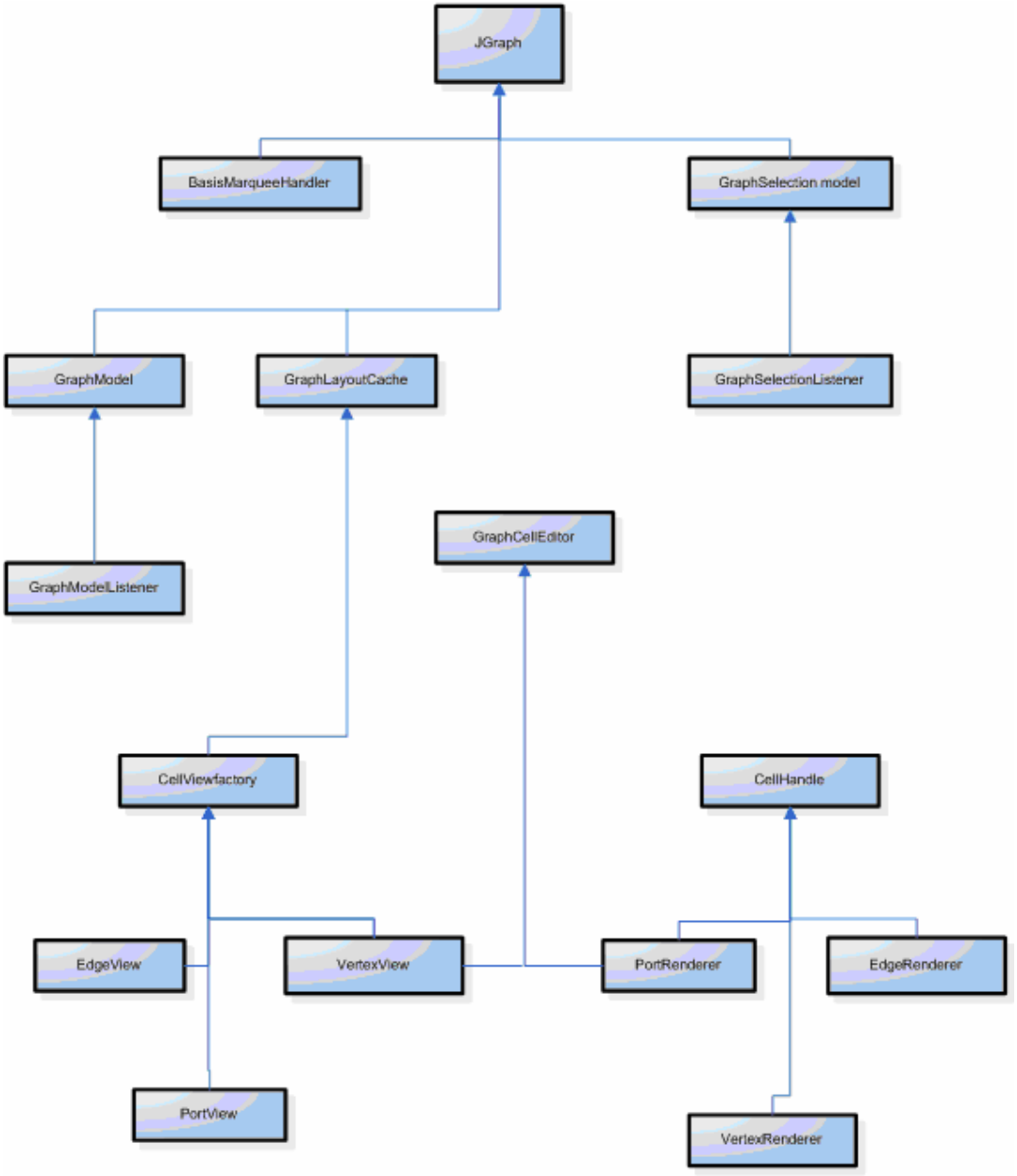
Quelques mots pour expliquer les démarches de l'intégration du logiciel. L'intégration d'une application graphique Java n'est pas très facile. Il existe plusieurs solutions, mais aucune n'est meilleure qu'une autre. Elles ont toutes des points forts et des points faibles. Dans notre projet nous avons choisi JGraph car il nous semble un outil adéquat et puisant pour notre projet.

Programmer avec Java en utilisant l'API externe JGraph pour la représentation graphique des réseaux

JGraph est un composant Java qui facilite le développement de représentations graphiques de réseau grâce aux principes de la théorie des graphes. À l'aide de JGraph, on peut réaliser des projets allant d'un simple éditeur graphique à la carte d'un réseau informatique, en passant par un programme de recherche du plus court chemin. Pour manipuler les composants JGraph on doit posséder une connaissance pratique de la théorie des graphes et de Java Swing. Un graphe est un cas particulier d'arbre où les noeuds peuvent posséder plusieurs parents disjoints ou pas.

- L'API Swing ne propose pas par défaut de composant permettant de représenter des graphes
- Utilisation de l'API externe JGraph
- Site : www.jgraph.com
- Elle fonctionne suivant un principe commun aux composants JTable et JTree dans le sens où elle se fonde sur l'architecture MVC
- De nombreux outils utilisent cette bibliothèque dont la licence est libre d'utilisation sous condition de fournir les sources

Voici une présentation graphique de JGraph



Programmation des fonctionnalités de zooming

```
public JToolBar createToolBar() {
            JToolBar toolbar = new JToolBar();
            toolbar.setFloatable(false);


            // Zoom Std
            URL zoomUrl = getClass().getClassLoader().getResource(
                        "resources/zoom.gif");
            ImageIcon zoomIcon = new ImageIcon(zoomUrl);
            toolbar.add(new AbstractAction("Zoom", zoomIcon) {
                  public void actionPerformed(ActionEvent e) {
                        myGraph.setScale(1.0);
                  }
            });

            // Zoom In
            URL zoomInUrl = getClass().getClassLoader().getResource(
                        "resources/zoomin.gif");
            ImageIcon zoomInIcon = new ImageIcon(zoomInUrl);
            toolbar.add(new AbstractAction("", zoomInIcon) {
                  public void actionPerformed(ActionEvent e) {
                        myGraph.setScale(2 * myGraph.getScale());
                  }
            });

            // Zoom Out
            URL zoomOutUrl = getClass().getClassLoader().getResource(
                        "resources/zoomout.gif");
            ImageIcon zoomOutIcon = new ImageIcon(zoomOutUrl);
            toolbar.add(new AbstractAction("", zoomOutIcon) {
                  public void actionPerformed(ActionEvent e) {
                        myGraph.setScale(myGraph.getScale() / 2);
                  }
            });


            return toolbar;
      }
```

Programmation de la fonctionnalité d'ouvrir d'un fichier XML

```
public void openFile() {
  int valeurRetour = JFileChooser.CANCEL_OPTION;
  initFileChooser();
  valeurRetour = fileChooser.showOpenDialog(null);
  if (valeurRetour == JFileChooser.APPROVE_OPTION) {
  myGraph = (new SaxJava().loadXML(fileChooser.getSelectedFile()));
 // MouseListener doubleClick de la sourie
  myGraph.addMouseListener(new MouseAdapter() {
   public void mousePressed(MouseEvent e) {
     if (e.getClickCount() == 2) {
          //position de la cellule
          int x = e.getX(), y = e.getY();
           Object cell = myGraph.getFirstCellForLocation(x, y);
          //    Info de la cellule
            if (cell != null) {
              DefaultGraphCell cell2 = new DefaultGraphCell();
               cell2 = (DefaultGraphCell)cell;
                    String lab1 =
                    myGraph.convertValueToString(cell2.getAttributes().get("ip"));
                    String lab2 =
                    myGraph.convertValueToString(cell2.getAttributes().get("ipconnect"
                    ));
                    String lab3 =
                    myGraph.convertValueToString(cell2.getAttributes().get("addMac"))
                    ;
                    String lab4 =
                    myGraph.convertValueToString(cell2.getAttributes().get("typeConn
                    ect"));
                    String affichage = "Addresse IP:  "+lab1 +"\n" +"Addresse connection:
                    "+lab2 +"\n" + "Addresse MAC:        "+lab3 +"\n" +
                    "Type de connection: "+lab4 +"\n";



          JOptionPane.showMessageDialog(null, affichage,"Information sur un
noeud",JOptionPane.CLOSED_OPTION);
           System.out.println(affichage);
           }
          }
}
});

insererFrame();
}
```

```
}
```

Programmation de la fonctionnalité d'utilisation d'image

```
public Icon image(String typeIcon){
        if( typeIcon.compareTo("server") == 0){
                return new ImageIcon( "resources/server03.gif" );
        }else{
                if( typeIcon.compareTo("computer") == 0)
                        return new ImageIcon( "resources/pc05.gif" );
                else
                        return  new ImageIcon( "resources/routeur.gif" );
        }
   }
```

Conclusion et recommandations :

- Nous avons réussit à développer une application de visualisation graphique de réseaux qui s'utilisera comme un module complémentaire du logiciel Nmap

- Ce projet nous a permis d'améliorer nos connaissances sur la programmation des graphes et nos connaissances en réseaux.

- Plusieurs extensions futures du logiciel peuvent être envisagées :
  - Améliorer les fonctionnalités de zooming
  - Programmation de la fonctionnalité d'impression
  - Etc.

Remerciement :

- On tient à remercier toutes les personnes qui nous ont soutenu tout au long de notre cheminement.

- On tient à remercier tous ceux qui ont assisté à notre présentation, Monsieur Michal Iglewski, Directeur du Département d'informatique et d'ingénierie, Monsieur Alain Charbonneau, Directeur du Module informatique.

- Finalement, nous tenons à remercier énormément notre superviseur Dr. Kamel Adi qui nous a apporté son soutien et son expertise dans le cadre du projet de synthèse.

Bibliographie :

Sites web consultés :

XUL
http://fr.wikipedia.org/wiki/XUL
http://www.mozilla.org/xpfe/xulref-french/

JGRAPH (java)
http://mbaron.developpez.com/javase/javavisu/
http://mondeca.wordpress.com/tag/decrire/

Nmap
http://www.nmap-tutorial.com/
http://fr.wikipedia.org/wiki/Nmap
http://www.tux-planet.fr/blog/?2006/12/15/132-utilisation-de-nmap-et-outil-de-detection-des-scans-de-ports
http://www.haypocalc.com/wiki/Nmap

Umit
http://umit.sourceforge.net/blog_redirect_en.html

Livres utilisés :

- Deitel & Deitel (Comment programmer en Java) Quatrième édition
- Peter (Mieux programmer en Java)