In [15]:

```
from pathlib import Path
import json
import spacy
import numpy as np
import networkx as nx
from pyvis.network import Network
import pandas as pd

from pdf_annotation import DataNode, TreeWidget, App
```

In [2]:

```
# load in an NLP model for demonstration
nlp = spacy.load("en_core_web_lg")
```

In [4]:

```
def loadf(path):
    """load a json file into a python dict"""
    with open(path) as f:
        data = json.load(f)
    return data
```

In [5]:

```
# load the digital representation of all processed documents created using the PdfDigitizer
doctrees = [DataNode(loadf(x)) for x in Path("C:\git\PDFDigitizer\pdfs").rglob("*.json")]
```

Indexing

Here, is an example of indexing a **DataNode** object. `doctrees` is a list of DataNodes. Each element represents the entire text of a pdf document.
The following cells use the `2020abrams-m1a2.pdf` as an example to...

1. Summarize the document heirarchy (as encoded via the PdfDigitizer tool) in the form of a **Table of Contents**, naturally.
2. Perform **DepthFirstSearch** to return and print out the section titled "Recommendations"

In [6]:

```
doctrees[0] # the __repr__ function returns a plain-text table of contents
```

Out[6]:

```
../pdfs/ARMY/2020abrams-m1a2.pdf
  Executive Summary
  System
    Abrams M1A2 System Enhancement Packages
    Trophy Active Protection System
  Mission
  Major Contractors
  Activity
    Abrams M1A2 System Enhancement Packages
    Trophy Active Protection System
  Assessment
    Abrams M1A2 System Enhancement Packages
    Trophy Active Protection System
  Recommendations
```

In [7]:

```
path, subtree = doctrees[0].search("Recommendations")
print(subtree.to_string())
```

```
. Evaluate the survivability of the Abrams SEPv3 with
Trophy APS against the most stressing threats identified by
the Intelligence Community .

. Develop operationally relevant requirements for the Abrams
M1A2 tank with and without the Trophy APS.

. Continue to develop and advance the appropriate modeling
and simulation tools needed to support the test planning and
evaluation of systems equipped with APS.

. Consider the findings of the DOT&E and Army LFT&E
SEPv3 evaluation reports to enhance the survivability of
future Abrams tank upgrades
```

Mission Similarity

The documents we have are all very similarly structured, they all have sections titled "Executive Summary", "Activity", "Mission", "Recommendations",...
In the cells below, we will utilize the indexing capability to compare the "Mission" specified in each of the documents.
The comparison metric is essentially a bag of words analysis. The structure we made using the PdfDigitizer allows for precise application to include only the sections titled "Mission"
An edge is drawn between any 2 documents which acheive a Mission-similarity score > %95.
Though, a more useful approach might involve a clustering algorithm to find cliches.

In [ ]:

```
system_descriptions = [doc.search("Mission") for doc in doctrees] # get the node titled "Mission" from each document
docs = [nlp(node.to_string()) for path, node in system_descriptions if node is not None] # use spacy to process all of the text wit
hin the nodes
N = len(docs)

edgelist = []

for i in range(N):
    for j in range(i+1, N):
        if docs[i].similarity(docs[j]) > .95:
            edgelist.append({"source":doctrees[i].label, "target":doctrees[j].label})

# Draw the network
df = pd.DataFrame(edgelist)
g = nx.from_pandas_edgelist(df, source='source',target='target')
net = Network(notebook=True)
net.from_nx(g)
net.show_buttons(filter_=['physics'])
net.show("doc_network.html")
```

In [30]:

```
# App("../pdfs/DOD/2020f35jsf.pdf") # Opens a pdf in the digitizer tool
```

Contractors

This section utilizes spacy's entity recognition model to extract the organizations refered to within the Major Contractors section of each document

In [41]:

```
system_descriptions = [doc.search("Major Contractors") for doc in doctrees] + [doc.search("Major Contractor") for doc in doctrees]
docs = [nlp(" ".join(node.to_string().replace("¢","").replace("\n", ".").split())) for path, node in system_descriptions if node is
not None]
```

In [45]:

```
contractors = set()

for doc in docs:
    for ent in doc.ents:
#         print("#"*100)
#         print(doc)
#         print("-"*50)
        if ent.label_ == "ORG":
#             print(str(ent))
            contractors.add(str(ent))
```

In [46]:

```
sorted(contractors)[:25]
```

Out[46]:

```
['Accenture Federal Services',
 'Anniston Army Depot',
 'BAE Systems',
 'BAE Systems Land and Armaments',
 'Bell Helicopter',
 'Bell-Boeing Joint Venture',
 'Boeing Defense, Space',
 'Boeing Helicopter Company',
 'Chemring Sensors and Electronic Systems',
 'Collins Aerospace',
 'DISA',
 'DRS Sustainment Systems',
 'DRS/Rafael – St. Louis',
 'Defensive Systems',
 'Electronic Systems',
 'Fincantieri Marinette Marine Corporation',
 'Fire Control',
 'GPS Source Inc.',
 'General Dynamics Information Technology',
 'General Dynamics Land Systems',
 'General Dynamics Marine Systems Bath Iron Works',
 'General Dynamics Mission',
 'General Dynamics Mission Systems',
 'General Electric Aviation – Evendale',
 'General Motors Defense']
```

In [ ]: