OVERVIEW

This document provides a final report for a project with the Center for Computational Learning Systems, Columbia University under the supervision of Prof. Gail Kaiser. This project is being undertaken as a 3-credit course under section 14 of COMS E6901 “Projects in Computer Science”.

The CCLS has an on-going project for ConEdison, which involves using machine learning algorithms to rank various feeders, transformers, and cables according to their rate of susceptibility to failure. These tools are used by Con Edison employees to better understand the components of a feeder (cables, joints, terminators, & transformers) that are susceptible to failure. This helps them better track down the root causes of failures and to use susceptibility to failure to guide how the system operates during periods of stress and guide replacement of weak components. My work this semester involved researching available cable data, to find replaced structures in the feeder hierarchies and building a model to collate joint information after mining available cable data.

PROJECT DESCRIPTION

My project this semester was concerned with two main categories of work, as described in this section.

Replaced Artifact Discovery

The first category of work involved investigating and studying the use of replacement and failure data from CAJAC to come up with records of cables and joints that have been replaced both the recent past and historically. The use of this data is for a higher fidelity representation of components changed for machine learning than what we have had before, which was yearly updates of assets. In addition to that, the potential fusion of data from other sources like Splice tickets, Vision cable snapshots that we get once or twice a year, weekly cable section data dump was investigated. The CAJAC database (please refer to Fig. 1), which is the main source for replaced artifact discovery, is available on VanSpliceSQL at ConEdison and copied at CCLS. It provides regular failure reports and
these reports are used to get an estimate of which cables and joints terminators and to a lesser degree transformers have been replaced due to failures only. Unfortunately, CAJAC does not record what the failed component is replaced with – this has to come from other data sources like the Splice Ticket database describe below.

With regards to this component of work, the various tables available in CAJAC were analyzed and Cajac dbo.Cable_Joint_failure was isolated as the major source of failure data from which recently replaced cables and joints can be determined. The table contains joints and cables in the “from Structure” (available as a composite Structure_A_Type and Structure_A_ID) and “to Structure” format to describe the starting manhole and ending manhole for a cable. These data points can be used to get basic cable and joint information including feeder id, network id and leg (where a leg refers to smaller sized cables that Con Edison splits a cable into, so that they fit in ducts; these may run for several manhole to manhole extents). The artifact is identified as a cable or joint by referring to the Equipment_Type field. Once this replacement data is obtained from CAJAC, joins can be taken, with the jeoJoints table (or with the inferred joints table produced in the other component of this project) available in Attributes in ediprodb1, for joint data and with Contingency_Cable_Data_History table, available in VanSpliceData in ediprodb2 for cable data.

In addition to this, further data fusion can be used to obtain the goal of a more complete and accurate collection of data on replaced artifacts. To this end, one of the major sources available is Splice ticket data that can be found in VanSpliceSQL. This database provides a record of all Splice tickets which are essentially a collection where all Joint replacement records are registered. From here, we can obtain data on all joints for which Splice tickets were registered and use this to fuse into the collected data from CAJAC. The table of interest here is Splice_Data, which contains full logs of the splice replacements.

Another important resource is the Contingency_Cable_Data_History, which can be used by virtue of the fact that regular updates of cable data are made available in this table since fall 2008. This database provides a weekly glimpse for cables and joints (using joint inference) that have changed. Joints inferred from this table can serve the same purpose for corroborating joint data. Furthermore, taking advantage of the historical nature of this data, deltas can be taken between timestamps on the cable data, and these deltas can be used to confirm the completeness of replaced artifact data.

**Artifact Discovery Strategy based on Investigation**

Based on the above discussion, and my study of the various resources available, I am able to propose the following outline, which defines a plan of action that can be undertaken to construct a data fusion based replaced artifact discovery application:

1. The first step would be to build a primary archive of replaced joints and cables. To build this archive, we will need to take deltas in relevant tables for joints and cables, depending on the time range for which the artifact discovery is required. For cables,
depending on the required dates, we can either go to contingency cable history or to Vision data. For joints, the best strategy would be to use the Joint Inference engine and generate joint records for given dates, then take deltas between them to get the primary replaced joint archive for the given time range. In addition to this, other resources for building the collection of replaced joints are the historical Jeopardy Joint data available in Attributes on ediprodb1 and on the Postgres database, ConEd_raw database on Gauss, and joints available from Con Ed’s IR department running their joint inference program upon request.

2. The next step is to visit the CAJAC database, especially, the Cable_Joint_Failure table which contains failure data for both cables and joints. This will help in confirming that some of cables and joints found as replaced, had failed and corroborate the discovered data.

3. Once this collection is built, the next step would be to further augment it with data fusion. An important source for this is Splice tickets for which data can be taken from the Splice_Data table and an outer join can be taken on this table with the Joint archive. All joints that are found in both tables with different installation dates support that these joints have been replaced. All new joints from Splice are added to the dated joint collection.

The following flow diagram depicts the above steps that need to be carried out for replaced artifact discovery:
Joint Inference

The second component involved coming up with a model to generate a complete set of joints by using various sources of data and collating them into a single joint inference resource. A major source among these were the various cable data sources, including Contingency history data, Vision data and cable data from jeopardy; and the existing joint
inference logic as well as past inferred-joint tables. Another resource will be the Splice Ticket data that includes information regarding historical joint splices, which are also available on VanSpliceSQL at Con Edison and copied at CCLS. A third resource will be FOWR data which is a work repository for trouble tracking (please refer to Fig. 1 for more information on all of these data sources). These resources will be utilized to collate and mine data from to create a possibly exhaustive set of joint data.

With regards to this component, extensive analysis was done on the available cable information that is available weekly. The data cable history is updated on a weekly basis in ediprodb2 database server, in VanSpliceData.dbo.Contingency_Cable_Data_History. The date on which the update was made is stored in the copy_date field. This is the major source from which joint inference will be carried out. The method to carry this out can be described in a nutshell as follows: All the distinct structures available are first determined. Once this is done, an algorithm is employed to find out all the cables coming in and going out of that structure (that is, for which cables a structure is a “from structure” and for which ones it is a “to structure”). Once this is known, it is matched to one of the following cable configurations [1] (shown partially, with only the first set of possible inferred joints, additional joints, up to 3 types, are also inferred for a given manhole code in some cases, the code takes them all into account):

<table>
<thead>
<tr>
<th>CODE</th>
<th>YEAR FROM</th>
<th>YEAR TO</th>
<th>NUMB ER1</th>
<th>TYPE1</th>
<th>GENERIC_NAME1</th>
<th>CONFIG 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>P3S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3S1</td>
<td>1985</td>
<td>1993</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3S1</td>
<td>1994</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink (rubber)</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>1W-1W</td>
</tr>
<tr>
<td>P1S1</td>
<td>1985</td>
<td>1999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
</tr>
<tr>
<td>P1S1</td>
<td>2000</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
</tr>
<tr>
<td>P3P3S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-1W</td>
</tr>
<tr>
<td>P3P3S1</td>
<td>1985</td>
<td>9999</td>
<td>2</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3P1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-1W</td>
</tr>
<tr>
<td>P3P1S1</td>
<td>1985</td>
<td>1991</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3P1S1</td>
<td>1992</td>
<td>1999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink (rubber)</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3P1S1</td>
<td>2000</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink (rubber)</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P1P1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-1W</td>
</tr>
<tr>
<td>P1P1S1</td>
<td>1985</td>
<td>1991</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-1W</td>
</tr>
<tr>
<td>P1P1S1</td>
<td>1992</td>
<td>1999</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-1W</td>
</tr>
<tr>
<td>P1P1S1</td>
<td>2000</td>
<td>9999</td>
<td>1</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>2W-1W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-3W</td>
</tr>
<tr>
<td>P3S1S1</td>
<td>1985</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P1S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-1W</td>
</tr>
<tr>
<td>P1S1S1</td>
<td>1985</td>
<td>1999</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-1W</td>
</tr>
<tr>
<td>P1S1S1</td>
<td>2000</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
</tr>
<tr>
<td>P3P3P3S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-1W</td>
</tr>
<tr>
<td>P3P3P3S1</td>
<td>1985</td>
<td>9999</td>
<td>3</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3P3P1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-1W</td>
</tr>
<tr>
<td>P3P3P1S1</td>
<td>1985</td>
<td>1991</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P3P3P1S1</td>
<td>1992</td>
<td>1999</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P3P1P1S1</td>
<td>2000</td>
<td>9999</td>
<td>2</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3P1P1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P3P1P1S1</td>
<td>1985</td>
<td>1991</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P3P3S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-1W</td>
</tr>
<tr>
<td>P3P3S1S1</td>
<td>1985</td>
<td>9999</td>
<td>2</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P3P1S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-3W</td>
</tr>
<tr>
<td>P3P1S1S1</td>
<td>1985</td>
<td>1999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P1P1S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P1P1S1S1</td>
<td>1985</td>
<td>1999</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P1P1S1S1</td>
<td>2000</td>
<td>9999</td>
<td>2</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
</tr>
<tr>
<td>P3S1S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>4W-3W</td>
</tr>
<tr>
<td>P3S1S1S1</td>
<td>1985</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
</tr>
<tr>
<td>P1S1S1S1</td>
<td>1900</td>
<td>1984</td>
<td>1</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P1S1S1S1</td>
<td>1985</td>
<td>1999</td>
<td>1</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-2W</td>
</tr>
<tr>
<td>P1S1S1S1</td>
<td>2000</td>
<td>9999</td>
<td>1</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
</tr>
</tbody>
</table>

This entire set of rules has been entered into a table called JointInferenceRules available in Assets in ediprodb1. The code picks up the rules from this database and the structures
from contingency data are matched against every rule to infer joints as described in the next paragraph.

Here, in the code field, each letter represents a cable type, P for paper type cables and S for solid cables, with the number next to the letter representing the number of conductors in the cable. The “Year From” and “Year To” represent the year range in which the cables were laid in the ground. This is utilized to find the type and number of joints in the manhole. For example, a configuration of type “P3P3S1” represents a manhole which has the ends of two three conductor paper cables and one single solid cable-set (Con Ed accounts for 3 separate cables for each phase as 1 cable-set in their data and analysis). If among the cables discovered for a manhole structure (as described in the previous paragraph), the youngest cable is in the year range 1900 to 1984, we can surmise that the manhole has a Lead Wipe type joint of configuration 4 way – 1 way and that it is a stop joint (the relevant row is highlighted in green above). There are some cases in which more complex joint configurations with different types of joints are employed (for instance, where multiple different types of joints can be inferred from a particular manhole), depending on date ranges, and these are also being inferred. This is essentially how joint inference is being carried out on the cable history data. The following table is a sample of some of the joints inferred for structures of that match the SQL “like” query string “M112%”:

<table>
<thead>
<tr>
<th>Struct</th>
<th>Feeder</th>
<th>Joint Type</th>
<th>Generic Name</th>
<th>Config</th>
<th>Stop</th>
<th>NetworkId</th>
</tr>
</thead>
<tbody>
<tr>
<td>M112</td>
<td>9B01</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>9B</td>
</tr>
<tr>
<td>M112</td>
<td>9B02</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>9B</td>
</tr>
<tr>
<td>M1120</td>
<td>5B21</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>5B</td>
</tr>
<tr>
<td>M11205</td>
<td>9B06</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>9B</td>
</tr>
<tr>
<td>M11208</td>
<td>23M68</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>23M</td>
</tr>
<tr>
<td>M11209</td>
<td>6Q33</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6Q</td>
</tr>
<tr>
<td>M1121</td>
<td>4X46</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>4X</td>
</tr>
<tr>
<td>M1121</td>
<td>5B22</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>5B</td>
</tr>
<tr>
<td>M11210</td>
<td>6Q81</td>
<td>Lead Wipe</td>
<td></td>
<td>1W-1W</td>
<td>N</td>
<td>6Q</td>
</tr>
<tr>
<td>M11211</td>
<td>23M66</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>23M</td>
</tr>
<tr>
<td>M11211</td>
<td>9B07</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-1W</td>
<td>N</td>
<td>9B</td>
</tr>
<tr>
<td>M11214</td>
<td>23M65</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>23M</td>
</tr>
<tr>
<td>M11214</td>
<td>23M65</td>
<td>Elastimold</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>23M</td>
</tr>
<tr>
<td>M11218</td>
<td>6Q37</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>6Q</td>
</tr>
<tr>
<td>M11218</td>
<td>6Q37</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>6Q</td>
</tr>
<tr>
<td>M11218</td>
<td>6Q37</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>6Q</td>
</tr>
<tr>
<td>M1122</td>
<td>5B22</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>5B</td>
</tr>
<tr>
<td>M11228</td>
<td>6Q31</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6Q</td>
</tr>
<tr>
<td>M1123</td>
<td>5B27</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>5B</td>
</tr>
<tr>
<td>M1123</td>
<td>9522</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>M1123</td>
<td>9B03</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>3W-1W</td>
<td>S</td>
<td>9B</td>
</tr>
<tr>
<td>M11232</td>
<td>6Q31</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>6Q</td>
</tr>
<tr>
<td>M11234</td>
<td>1Q18</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>1Q</td>
</tr>
<tr>
<td>M11236</td>
<td>6B52</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11236</td>
<td>6B52</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>------</td>
<td>------------</td>
<td>-------</td>
<td>-----</td>
<td>----</td>
</tr>
<tr>
<td>M11237</td>
<td>6B51</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11237</td>
<td>6Q33</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6Q</td>
</tr>
<tr>
<td>M11238</td>
<td>6B52</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11238</td>
<td>6Q31</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6Q</td>
</tr>
<tr>
<td>M11242</td>
<td>5Q46</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11243</td>
<td>7Q66</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>7Q</td>
</tr>
<tr>
<td>M11244</td>
<td>7Q65</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>7Q</td>
</tr>
<tr>
<td>M11249</td>
<td>6B51</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11250</td>
<td>11M53</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11252</td>
<td>5Q43</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11253</td>
<td>5Q42</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11254</td>
<td>11M57</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
<td>S</td>
<td>11M</td>
</tr>
<tr>
<td>M11256</td>
<td>7362</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>M11258</td>
<td>7Q78</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>7Q</td>
</tr>
<tr>
<td>M11259</td>
<td>6B52</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11261</td>
<td>11M57</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>1W-1W</td>
<td>S</td>
<td>11M</td>
</tr>
<tr>
<td>M11261</td>
<td>11M57</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11261</td>
<td>5Q36</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>2W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11261</td>
<td>5Q37</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>5Q</td>
</tr>
<tr>
<td>M11261</td>
<td>5Q37</td>
<td>RAYCHEM</td>
<td>Heat shrink</td>
<td>3W-1W</td>
<td>S</td>
<td>5Q</td>
</tr>
<tr>
<td>M11261</td>
<td>5Q37</td>
<td>Elastimold</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11262</td>
<td>6B51</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>2W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11264</td>
<td>11M47</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11265</td>
<td>5Q37</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>3W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11265</td>
<td>5Q37</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>3W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11266</td>
<td>7Q79</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>7Q</td>
</tr>
<tr>
<td>M11268</td>
<td>6Q81</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>1W-1W</td>
<td>N</td>
<td>6Q</td>
</tr>
<tr>
<td>M11270</td>
<td>11M49</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11270</td>
<td>11M57</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11270</td>
<td>5Q43</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11270</td>
<td>6B52</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11272</td>
<td>5Q43</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>5Q</td>
</tr>
<tr>
<td>M11275</td>
<td>11M49</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11275</td>
<td>11M57</td>
<td>RICHARDS</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11275</td>
<td>6B52</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11278</td>
<td>11M54</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11278</td>
<td>11M54</td>
<td>LW</td>
<td>Lead Wipe</td>
<td>1W-1W</td>
<td>N</td>
<td>11M</td>
</tr>
<tr>
<td>M11279</td>
<td>6B51</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>6B</td>
</tr>
<tr>
<td>M11283</td>
<td>9B08</td>
<td>ELASTIMOLD</td>
<td>Disconnectable</td>
<td>1W-1W</td>
<td>N</td>
<td>9B</td>
</tr>
</tbody>
</table>

Outline for Augmenting Inference Engine

In addition to the contingency history data, data from FOWR and Splice tickets, as described in the previous section, can also be used in a similar manner to bolster the inferred details from contingency data. With regards to these other sources the following
Outline has been devised to augment the Inference engine to use data from other sources and again, use data fusion to corroborate the accuracy of inferred joints.

1. The first step for augmenting the inference engine would be to go to Vision, as well as historical cable data in Jeopardy and older contingency history snap shots for getting older snapshots of cable data, so that the engine can deal with wider timeline requirements, and joints from older cable configurations can be inferred.

2. The next step would be to go to Splice tickets, where we have records of joints that have been replaced, and these can be used to confirm and update the set of joints inferred for a particular cable snapshot.

3. Finally, it is also possible to look into the FOWR trouble tracking repository where Stop joint data is available in the Stop_joints table in Contingency, in M02017rj.FOWR. Here, depending on the inspection status of the Stop joint recorded, the joints inferred can further be updated as this records actual field verification of stop joints.

The following diagram shows the various data sources available at Coned and the ones highlighted in green represent the ones that have been used or are in the process of being employed for the two components of work.
CONCLUSION

In conclusion, with regards to the first component, investigation was carried out into how to effectively use the available resources and employ data fusion upon them, to build a plan for discovering replaced artifacts. As a result of this investigation, a complete strategy has been devised to facilitate the building of an archive for replaced joints and cables, along with the data flow involved for such a process.

For the second component, a complete C# based Joint Inference engine has been written to use contingency cable history data, to infer joints and relevant details about them. Furthermore, a strategy has been outlined to pursue fusion on the inference data to use other sources and make this data more accurate and complete. The version controlled code is available at the following directory:

https://power.ldeo.columbia.edu/svn/Proj/ConEd/Apps/trunk/JointInference/

CODE

This section contains part of the JointInference engine: the C# code written for inferring joints using rules from the JointInferenceRules table in Assets:

```csharp
using System;
using System.Collections.Generic;
using System.Text;
using System.Data.SqlClient;
using System.Data;
using System.Collections;
using System.Configuration;
using System.Data.Common;
using System.Diagnostics;
namespace JointInferences
{
    public class JointInference
    {
        #region protected and private fields
        protected SqlConnection m_conn;
        protected SqlConnection m_conn1;
        protected SqlConnection m_conn2;
        #endregion

        public ArrayList joints = new ArrayList();

        public DataTable JointInferenceMain(string structure)
```
```java
{ 
    ArrayList alStructures = new ArrayList();
    ArrayList alTemp = new ArrayList();
    DataTable dtJoints = CreateJointsTable();

    /* DB connection */
    m_conn = new SqlConnection(ConfigurationManager.AppSettings["db2Conn"]);
    m_conn.Open();

    string sql = "SELECT DISTINCT From_Structure as Structure, Feeder_ID " +
                "FROM VanSpliceData.dbo.Contingency_Cable_Data_History " +
                "WHERE (Cable_Type = 'Solid' OR Cable_Type = 'Paper' " +
                "OR Cable_Type='EPR' OR Cable_Type='LCR' OR Cable_Type='RUBBER' " +
                "OR Cable_Type='XLP') AND CopyDate = (select max(CopyDate) from VanSpliceData.dbo.Contingency_Cable_Data_History) " +
                "AND From_Structure LIKE " + structure + "%" AND Conductors IS NOT NULL" +
                " UNION " +
                "SELECT DISTINCT To_Structure as Structure, Feeder_ID " +
                "FROM VanSpliceData.dbo.Contingency_Cable_Data_History " +
                "WHERE (Cable_Type = 'Solid' OR Cable_Type = 'Paper' " +
                "OR Cable_Type='EPR' OR Cable_Type='LCR' OR Cable_Type='RUBBER' " +
                "OR Cable_Type='XLP') AND CopyDate = (select max(CopyDate) from VanSpliceData.dbo.Contingency_Cable_Data_History) " +
                "AND To_Structure LIKE " + structure + "%" AND Conductors IS NOT NULL";

    SqlCommand cmd = new SqlCommand(sql, m_conn);
    SqlDataReader dr = cmd.ExecuteReader();

    if (dr.HasRows)
    {
        while (dr.Read())
        {
            inferJoints(((String)dr["Structure"]).Trim(),
                         ((String)dr["Feeder_ID"]).Trim(), dtJoints);
        }
    }

    dr.Close();
    m_conn.Close();
    return dtJoints;
}
```
public DataTable JointInferenceMain()
{
    ArrayList alStructures = new ArrayList();
    ArrayList alTemp = new ArrayList();
    DataTable dtJoints = CreateJointsTable();

    /* DB connection */
    m_conn = new SqlConnection(ConfigurationManager.AppSettings["db2Conn"]);
    m_conn.Open();

    string sql = "SELECT DISTINCT From_Structure as Structure, Feeder_ID " +
        "FROM VanSpliceData.dbo.Contingency_Cable_Data_History " +
        "WHERE (Cable_Type = 'Solid' OR Cable_Type = 'Paper' " +
        "OR Cable_Type='EPR' OR Cable_Type='LCR' OR Cable_Type='RUBBER' " +
        "OR Cable_Type='XLP') AND CopyDate = (select max(CopyDate) from VanSpliceData.dbo.Contingency_Cable_Data_History) "+
        "AND Conductors IS NOT NULL" +
        " UNION " +
        "SELECT DISTINCT To_Structure as Structure, Feeder_ID " +
        "FROM VanSpliceData.dbo.Contingency_Cable_Data_History " +
        "WHERE (Cable_Type = 'Solid' OR Cable_Type = 'Paper' " +
        "OR Cable_Type='EPR' OR Cable_Type='LCR' OR Cable_Type='RUBBER' " +
        "OR Cable_Type='XLP') AND CopyDate = (select max(CopyDate) from VanSpliceData.dbo.Contingency_Cable_Data_History)" +
        "AND Conductors IS NOT NULL";

    SqlCommand cmd = new SqlCommand(sql, m_conn);
    SqlDataReader dr = cmd.ExecuteReader();

    if (dr.HasRows)
    {
        while (dr.Read())
        {
            inferJoints(((String)dr["Structure"]).Trim(),
                ((String)dr["Feeder_ID"]).Trim(), dtJoints);
        }
    }

    dr.Close();
    m_conn.Close();

    return dtJoints;
}

public void inferJoints(String structId, String feederId, DataTable dtJoints)
```csharp
int p1 = 0;
int s1 = 0;
int p3 = 0;
int s3 = 0;
int year = 1971;
string networkId = "";
//Boolean temp = false;

/* DB connection */
m_conn1 = new SqlConnection(ConfigurationManager.AppSettings["db2Conn"]);
m_conn1.Open();

string sql = "SELECT * FROM VanSpliceData." +
"dbo.Contingency_Cable_Data_History WHERE (From_Structure = "" + structId +"
" OR To_Structure = "" + structId + ") AND Conductors IS NOT NULL" +
"AND CopyDate = (select max(CopyDate) from VanSpliceData.dbo.Contingency_Cable_Data_History) AND Feeder_ID = "" + feederId +
"AND (Cable_Type = 'Solid' OR Cable_Type = 'Paper' OR Cable_Type='EPR'"
+ " OR Cable_Type='LCR' OR Cable_Type='RUBBER' OR Cable_Type='XLP');"

SqlCommand cmd = new SqlCommand(sql, m_conn1);
SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)
{
    while (dr.Read())
    {
        if ((((String)dr["Cable_Type"]).Trim().Equals("Solid") ||
            ((String)dr["Cable_Type"]).Trim().Equals("SOLID") ||
            ((String)dr["Cable_Type"]).Trim().Equals("EPR") ||
            ((String)dr["Cable_Type"]).Trim().Equals("LCR") ||
            ((String)dr["Cable_Type"]).Trim().Equals("RUBBER") ||
            ((String)dr["Cable_Type"]).Trim().Equals("XLP")) && Convert.ToInt32(dr["Conductors"]) == 1)
        {
            s1++;
        }
    }
}
```
((String)dr["Cable_Type"]).Trim().Equals("EPR") ||
((String)dr["Cable_Type"]).Trim().Equals("LCR") ||
((String)dr["Cable_Type"]).Trim().Equals("RUBBER") ||
((String)dr["Cable_Type"]).Trim().Equals("XLP")) &&
Convert.ToInt32(dr["Conductors"]) == 3)
{
    s3++;
}
}

if (((String)dr["Cable_Type"]).Trim().Equals("Paper") ||
((String)dr["Cable_Type"]).Trim().Equals("PAPER")) &&
Convert.ToInt32(dr["Conductors"]) == 1)
{
    p1++;  
}

if (((String)dr["Cable_Type"]).Trim().Equals("Paper") ||
((String)dr["Cable_Type"]).Trim().Equals("PAPER")) &&
Convert.ToInt32(dr["Conductors"]) == 3)
{
    p3++;
}
}

if (!dr["Install_Date"].Equals(DBNull.Value))
{
    if (year < (Convert.ToDateTime(dr["Install_Date"]))).Year)
    {
        year = (Convert.ToDateTime(dr["Install_Date"]))).Year;
    }
}

networkId = ((String)dr["Network_Id"]).Trim();

/* DB connection */
m_conn2 = new SqlConnection(ConfigurationManager.AppSettings["db1Conn"]);
m_conn2.Open();

string sql2 = "SELECT * FROM Assets.dbo.JointInferenceRules";
SqlCommand cmd2 = new SqlCommand(sql2, m_conn2);
SqlDataReader dr2 = cmd2.ExecuteReader();
string code;
int s11 = 0, s33 = 0, p11 = 0, p33 = 0;

DataRow row, row1, row2;

if (dr2.HasRows)
{
    while (dr2.Read())
    {
        code = ((string)dr2["Code"]).Trim();
        s11 = 0;
        s33 = 0;
        p11 = 0;
        p33 = 0;

        for (int i = 0; i < code.Length; i=i+2)
        {
            if (code.Substring(i, 2).Equals("S1"))
            {
                s11++;   
            }
            if (code.Substring(i, 2).Equals("S3"))
            {
                s33++; 
            }
            if (code.Substring(i, 2).Equals("P1"))
            {
                p11++;     
            }
            if (code.Substring(i, 2).Equals("P3"))
            {
                p33++; 
            }
        }

        if (p1 == p11 && p3 == p33 && s1 == s11 && s3 == s33)
        {
            if (year >= ((int)dr2["Year_From"]) && year <= ((int)dr2["Year_To"]))
            {
                for(int i=0;i<((int)dr2["Number1"]);i++)
                {
                    row = dtJoints.NewRow();
                    row["Struct"] = structId;
                    row["Feeder"] = feederId;
                }
            }
        }
    }
}
row["Joint Type"] = ((string)dr2["Type1"]).Trim();
row["Generic Name"] = ((string)dr2["Generic_Name1"]).Trim();
row["Config"] = ((string)dr2["Config1"]).Trim();
row["Stop"] = ((string)dr2["Stop1"]).Trim();
row["NetworkId"] = networkId;
dtJoints.Rows.Add(row);
}

if (!dr2["Number2"].Equals(DBNull.Value))
{
    for (int i = 0; i < ((int)dr2["Number2"]); i++)
    {
        row1 = dtJoints.NewRow();
        row1["Struct"] = structId;
        row1["Feeder"] = feederId;
        row1["Joint Type"] = ((string)dr2["Type2"]).Trim();
        row1["Generic Name"] = ((string)dr2["Generic_Name2"]).Trim();
        row1["Config"] = ((string)dr2["Config2"]).Trim();
        row1["Stop"] = ((string)dr2["Stop2"]).Trim();
        row1["NetworkId"] = networkId;
        dtJoints.Rows.Add(row1);
    }
}

if (!dr2["Number3"].Equals(DBNull.Value))
{
    for (int i = 0; i < ((int)dr2["Number3"]); i++)
    {
        row2 = dtJoints.NewRow();
        row2["Struct"] = structId;
        row2["Feeder"] = feederId;
        row2["Joint Type"] = ((string)dr2["Type3"]).Trim();
        row2["Generic Name"] = ((string)dr2["Generic_Name2"]).Trim();
        row2["Config"] = ((string)dr2["Config3"]).Trim();
        row2["Stop"] = ((string)dr2["Stop3"]).Trim();
        row2["NetworkId"] = networkId;
        dtJoints.Rows.Add(row2);
    }
}
}
}
}
}
}

dr2.Close();
m_conn2.Close();

dr.Close();
m_conn1.Close();
}

private DataTable CreateJointsTable()
{
    DataTable dt = new DataTable();
    dt.Columns.Add("Feeder", System.Type.GetType("System.String"));
    dt.Columns.Add("Joint Type", System.Type.GetType("System.String"));
    dt.Columns.Add("Generic Name", System.Type.GetType("System.String"));
    dt.Columns.Add("Config", System.Type.GetType("System.String"));
    dt.Columns.Add("Stop", System.Type.GetType("System.String"));
    dt.Columns.Add("NetworkId", System.Type.GetType("System.String"));

    return dt;
}


REFERENCES

[1] These joint inference rules were produced by Albert Boulanger’s 2008 summer intern, Robin Broder.