In [3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter
import missingno as miss
```

Before starting we import various interesting different libraries, which we will use later.

In [4]:
```
import seaborn as sns
sns.set(style="whitegrid", palette="muted")
current_palette = sns.color_palette()
```

In [5]:
```
pd.set_option('display.max_columns', None)
```

Now we read our initial file from the dataset.

In [136]:
```
df = pd.read_csv('DatabaseBigData.csv')
```

In [137]:
```
df
```

Out[137]:

	Unnamed: 0	IMPACT	Likelihood To Recommend	Jan	Feb	Mar
0	Sample Size	32976	32976	17	17	367
1	Overall Satisfaction	0.070595052	0.070550124	-	-	-
2	Understandability of Information	0.036293045	0.036269947	-	-	-
3	Claims handling satisfaction	0.036210869	0.036187823	-	-	-
4	Support and assistance satisfaction	0.036109393	0.036086412	-	-	-
5	Speed of Requests	0.034805942	0.034783791	-	-	-

In [138]:
```
df.head(50)
```

Out[138]:

	Unnamed: 0	IMPACT	Likelihood To Recommend	Jan	Feb	Mar
0	Sample Size	32976	32976	17	17	367
1	Overall Satisfaction	0.070595052	0.070550124	-	-	-
2	Understandability of Information	0.036293045	0.036269947	-	-	-
3	Claims handling satisfaction	0.036210869	0.036187823	-	-	-
4	Support and assistance satisfaction	0.036109393	0.036086412	-	-	-
5	Speed of Requests	0.034805942	0.034783791	-	-	-

In [5]:
```
first_row = df['Unnamed: 0']
```

We scrape all of the names of the first column, in case we will need them later.

In [6]:
```
#first_row
```

Missing Data

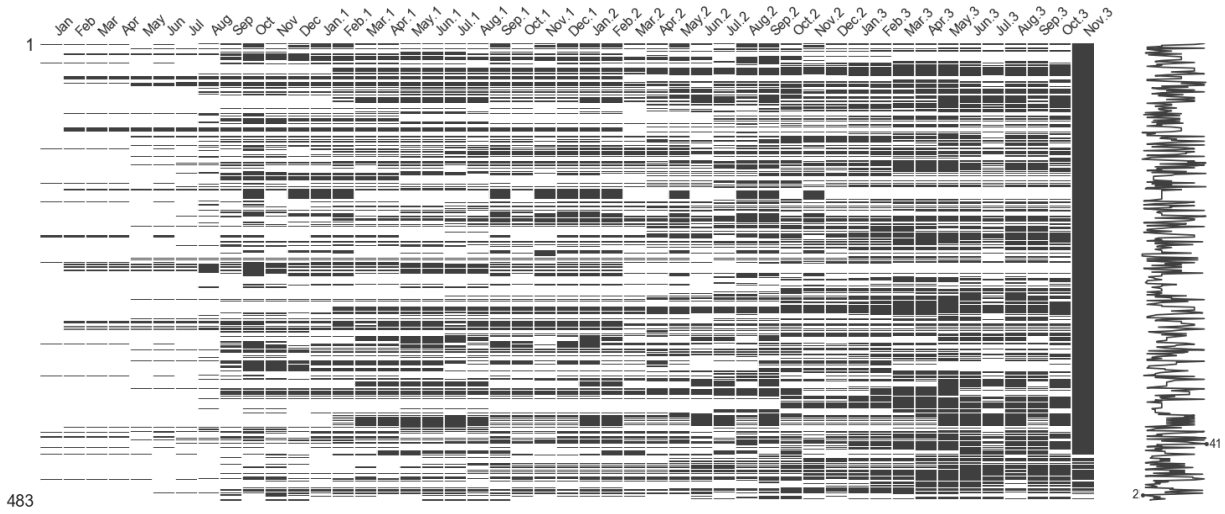
We no analyze the amount of missing data within the dataset, we will use in this case a dataset in which will delete all not directly survey-related has already been deleted.

The following matrix displays all the missing values that have been vacant in the dataset.

In [62]:
```
df2 = pd.read_csv('DriverAnalysisNT.csv')
del df2['Pain points']
del df2['Main category']
del df2['# Surveys']
del df2['Weight']
del df2['Overall']
del df2['Unnamed: 0']
```

In [63]:
```
miss.matrix(df2)
```

Out[63]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2660f0e29b0>
```



Data Preparation

We use now another transposed version of the dataset and scrape the names of the months time period.

In [7]:
```
dff = pd.read_csv('DriverAnalysisT.csv')
dff.head()
```

Out[7]:

	Unnamed: 0	Overall Satisfaction	Ability to reach required service	Ability to reach to regional sales team	Ability to resolve my issues	Accessibility to right person (Life)
0	Pain points	Overall Satisfaction	Speed of resolution	Speed of resolution	Value proposition	Speed of resolution
1	Main category	Overall Satisfaction	Speed of Resolution/service support	Speed of Resolution/service support	Value proposition/products	Speed of Resolution/service support
2	Jan	NaN	5.214285714	NaN	NaN	NaN
3	Feb	NaN	6.05	NaN	NaN	NaN
4	Mar	NaN	6.047619048	NaN	NaN	NaN

Now we scrape the names of the months within the time period.

In [8]:
```
first_row = dff['Unnamed: 0']
```

In [9]:
```
first_row
```

Out[9]:
```
0       Pain points
1     Main category
2               Jan
3               Feb
4               Mar
5               Apr
6               May
7               Jun
8               Jul
9               Aug
10              Sep
11              Oct
12              Nov
13              Dec
14              Jan
15              Feb
16              Mar
17              Apr
18              May
19              Jun
20              Jul
21              Aug
22              Sep
23              Oct
24              Nov
25              Dec
26              Jan
27              Feb
28              Mar
29              Apr
30              May
31              Jun
32              Jul
33              Aug
34              Sep
35              Oct
36              Nov
37              Dec
38              Jan
39              Feb
40              Mar
41              Apr
42              May
43              Jun
44              Jul
45              Aug
46              Sep
47              Oct
48              Nov
49          Overall
50        # Surveys
51           Weight
Name: Unnamed: 0, dtype: object
```

Now we start analyzing the numerical values of the datset more thoroughly. We import a cleaned version of the dataset, in which our group specified the membership of every single driver to a pain point and a main category. We thus gathered two new categorical columns. We further will analyze the effects that these and try to find useful pattern, which could help the company to understand clearer how to benefit from this information.

In [65]:
```
df2 = pd.read_csv('DriverAnalysisNT.csv')
df2.head()
```

Out[65]:

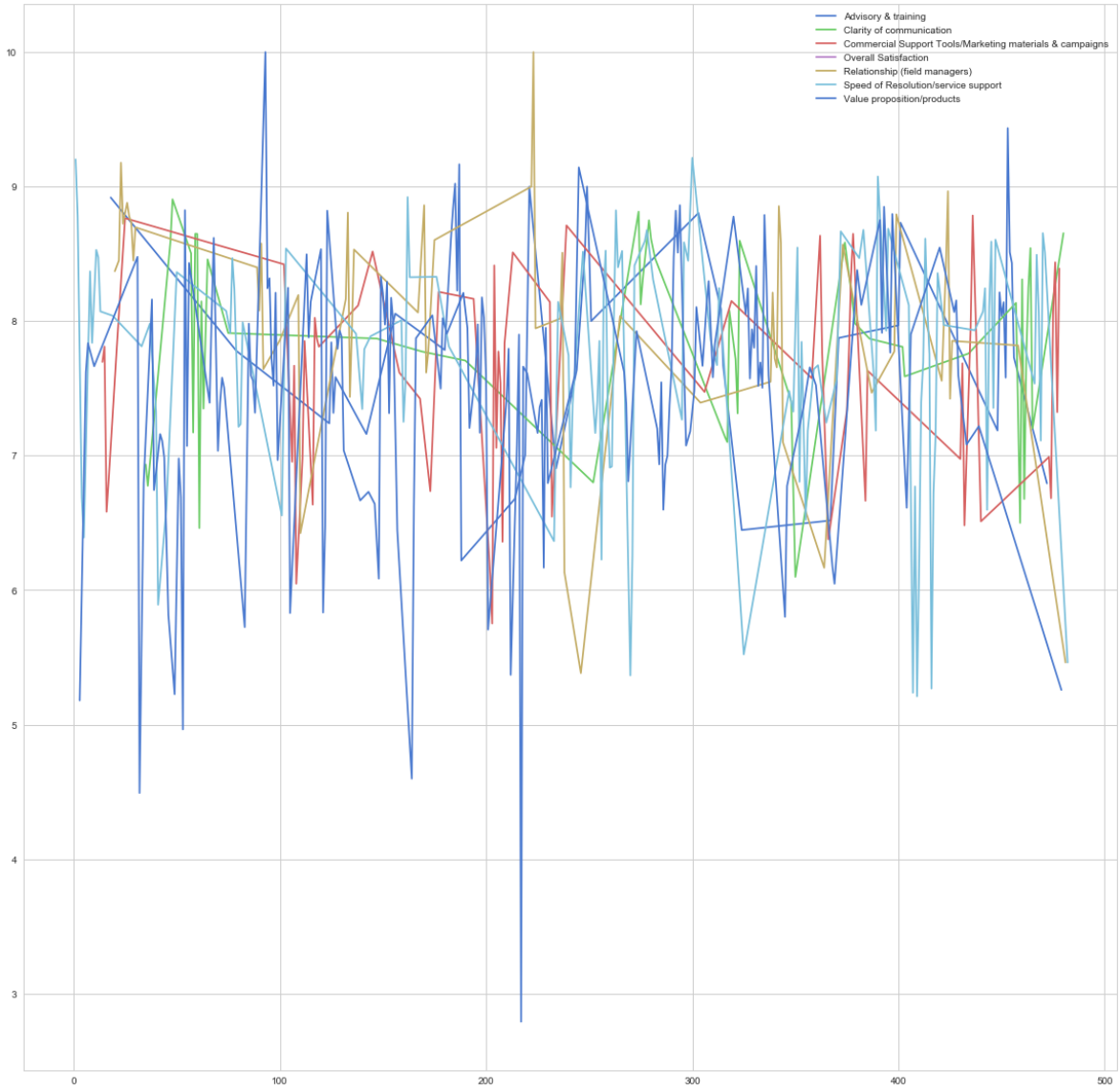
	Unnamed: 0	Pain points	Main category	Jan	Feb	Mar	Apr	May	Jun
0	Overall Satisfaction	Overall Satisfaction	Overall Satisfaction	NaN	NaN	NaN	NaN	NaN	NaN
1	Ability to reach required service	Speed of resolution	Speed of Resolution/service support	5.214286	6.05	6.047619	6.142857	NaN	6.163265
2	Ability to reach to regional sales team	Speed of resolution	Speed of Resolution/service support	NaN	NaN	NaN	NaN	NaN	NaN
3	Ability to resolve my issues	Value proposition	Value proposition/products	NaN	NaN	NaN	NaN	NaN	NaN
4	Accessibility to right person (Life)	Speed of resolution	Speed of Resolution/service support	NaN	NaN	NaN	NaN	NaN	NaN

To get a better understanding of the complexity of the different survey values, we plot their overall general values. Our goal is to find patterns in the following graphs. As we can see, our values are very complex and thus we are not able to detect any patterns from the data at first sight.

In [14]:
```
dfx = df2
%pylab inline
figsize(20,20)
dfx.groupby('Main category')['Overall'].plot(legend=True)
```

```
Populating the interactive namespace from numpy and matplotlib
```

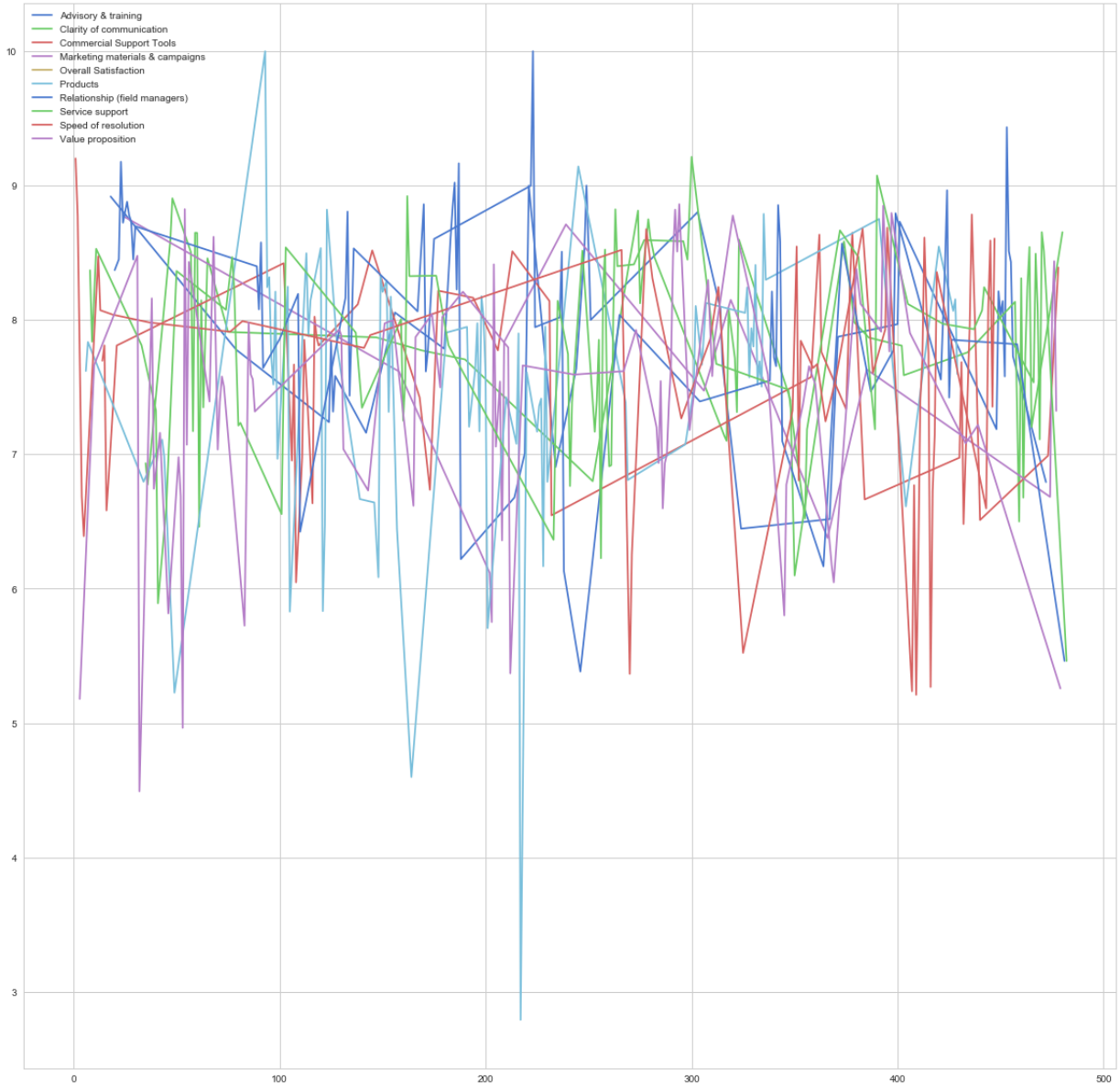
Out[14]:
```
Main category
Advisory & training                                          AxesSubplot(0.125,
0.125;0.775x0.755)
Clarity of communication                                     AxesSubplot(0.125,
0.125;0.775x0.755)
Commercial Support Tools/Marketing materials & campaigns     AxesSubplot(0.125,
0.125;0.775x0.755)
Overall Satisfaction                                         AxesSubplot(0.125,
0.125;0.775x0.755)
Relationship (field managers)                                AxesSubplot(0.125,
0.125;0.775x0.755)
Speed of Resolution/service support                          AxesSubplot(0.125,
0.125;0.775x0.755)
Value proposition/products                                   AxesSubplot(0.125,
0.125;0.775x0.755)
Name: Overall, dtype: object
```


Advisory & training
Clarity of communication
Commercial Support Tools/Marketing materials & campaigns
Overall Satisfaction
Relationship (field managers)
Speed of Resolution/service support
Value proposition/products
10
9
8
7
6
5
4
3
0
100
200
300
400
500

In [15]:
```
dfx = df2
%pylab inline
figsize(20,20)
dfx.groupby('Pain points')['Overall'].plot(legend=True)
```

```
Populating the interactive namespace from numpy and matplotlib
```

Out[15]:
```
Pain points
Advisory & training                AxesSubplot(0.125,0.125;0.775x0.755)
Clarity of communication           AxesSubplot(0.125,0.125;0.775x0.755)
Commercial Support Tools           AxesSubplot(0.125,0.125;0.775x0.755)
Marketing materials & campaigns    AxesSubplot(0.125,0.125;0.775x0.755)
Overall Satisfaction               AxesSubplot(0.125,0.125;0.775x0.755)
Products                           AxesSubplot(0.125,0.125;0.775x0.755)
Relationship (field managers)      AxesSubplot(0.125,0.125;0.775x0.755)
Service support                    AxesSubplot(0.125,0.125;0.775x0.755)
Speed of resolution                AxesSubplot(0.125,0.125;0.775x0.755)
Value proposition                  AxesSubplot(0.125,0.125;0.775x0.755)
Name: Overall, dtype: object
```



As our general plot was not very successful in gathering us a lot information regarding the patterns of the data, we now turn our focus now to descriptive statistics.

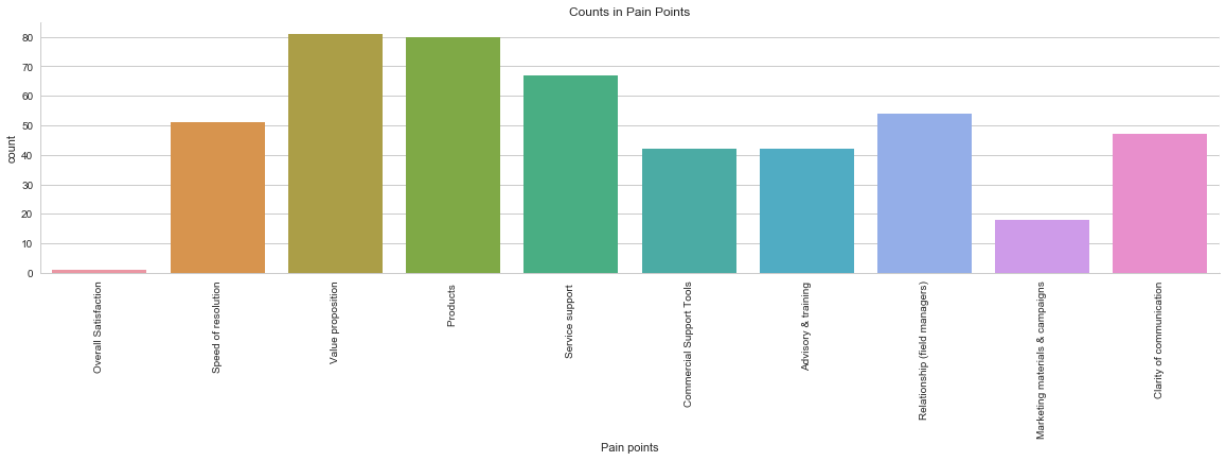
Descriptive Statistics

We now are interested in determining the number of counts for every single pain point and main category.

As we can see in the graphs, the 'Overall Satisfaction'-category has in both cases only one value, as it consists of only a single driver, which averages the scores received over the relative period of time.

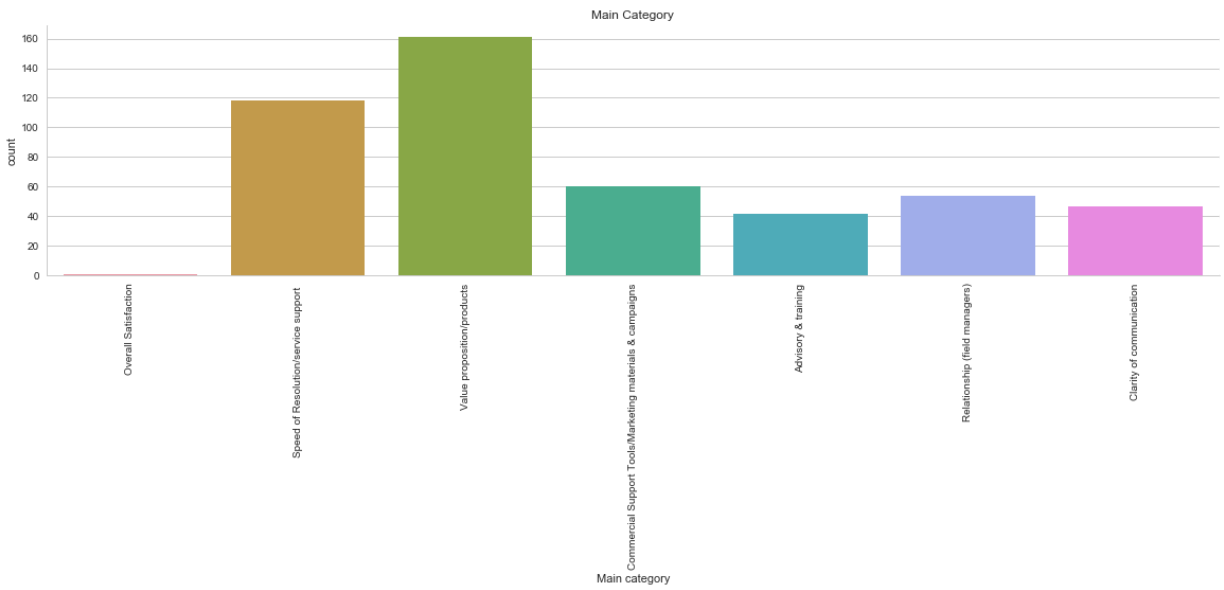
In [33]:
```
dfx = pd.read_csv('DriverAnalysisNT.csv')
%pylab inline
figsize(20,20)
g = sns.factorplot("Pain points", data=dfx, aspect=4, kind="count")
g.set_xticklabels(rotation=90)
g = plt.title("Counts in Pain Points")
```

```
Populating the interactive namespace from numpy and matplotlib
```



In [77]:
```
dfx = pd.read_csv('DriverAnalysisNT.csv')
%pylab inline
figsize(20,20)
g = sns.factorplot("Main category", data=dfx, aspect=4, kind="count")
g.set_xticklabels(rotation=90)
g = plt.title("Main Category")
```

```
Populating the interactive namespace from numpy and matplotlib
```



In numerical terms:

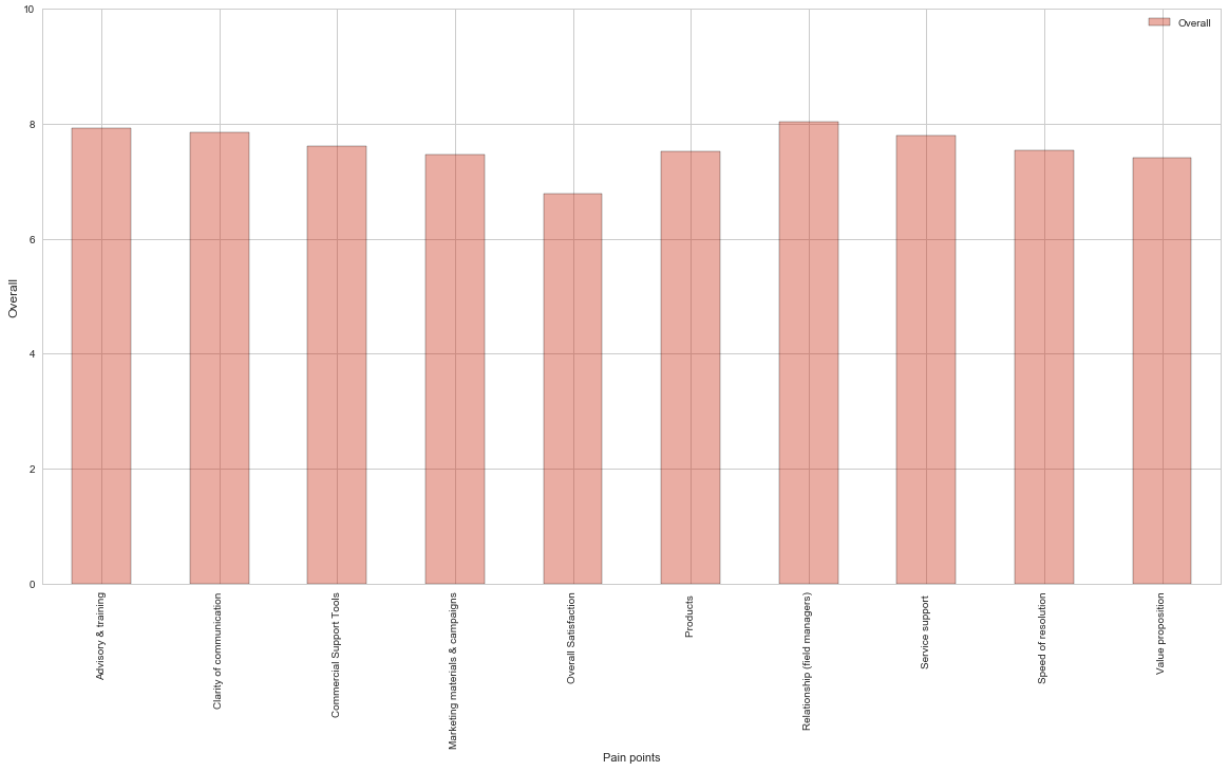
In [42]:
```
chars = ['Pain points', 'Main category']
for i in chars:
    print(pd.value_counts(df_[i]))
```

```
Value proposition                  81
Products                           80
Service support                    67
Relationship (field managers)      54
Speed of resolution                51
Clarity of communication           47
Advisory & training                42
Commercial Support Tools           42
Marketing materials & campaigns    18
Overall Satisfaction                1
Name: Pain points, dtype: int64
Value proposition/products                                    161
Speed of Resolution/service support                           118
Commercial Support Tools/Marketing materials & campaigns       60
Relationship (field managers)                                  54
Clarity of communication                                       47
Advisory & training                                            42
Overall Satisfaction                                            1
Name: Main category, dtype: int64
```

Now we inspect the mean and median values of our two categorical variables. Interestingly enough, the Overall Satisfaction variable scores in both cases the lowest, pointing towards a negative attitude of survey takers in regard to the score that they had given to the company, considering that every part of the company has scored higher than what is displayed in the Overall Satisfaction case.

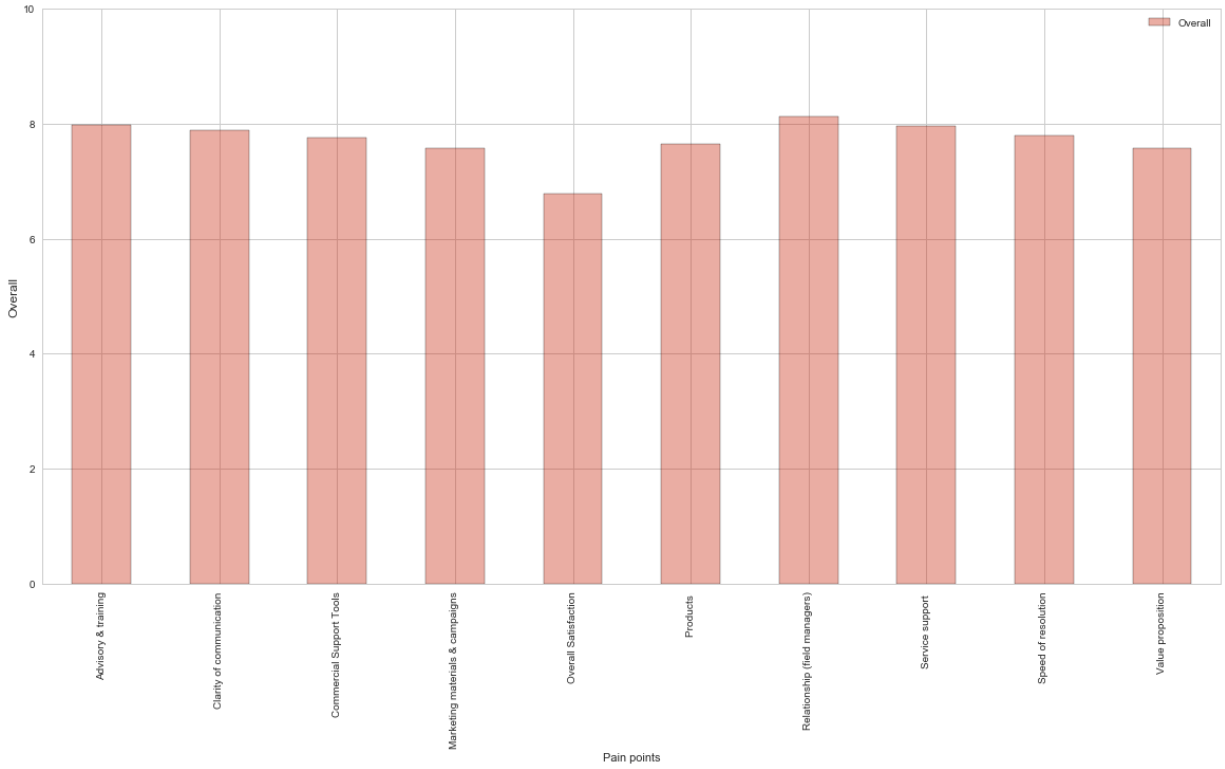
In [28]:
```
dfx = df2
dfx = dfx.groupby(['Pain points'])['Overall'].mean()
%pylab inline
figsize(20,10)
dfx.plot.bar(color=(0.8, 0.2, 0.1, 0.4),  edgecolor='black', legend=True)
plt.ylabel("Overall", size = 12)
ylim(0,10)
plt.savefig("PainpointsOverall.png", dpi=400)
```

```
Populating the interactive namespace from numpy and matplotlib
```



In [38]:
```
dfx = df2
dfx = dfx.groupby(['Pain points'])['Overall'].median()
%pylab inline
figsize(20,10)
dfx.plot.bar(color=(0.8, 0.2, 0.1, 0.4),  edgecolor='black', legend=True)
plt.ylabel("Overall", size = 12)
ylim(0,10)
plt.savefig("PainpointsOverall.png", dpi=400)
```

```
Populating the interactive namespace from numpy and matplotlib
```

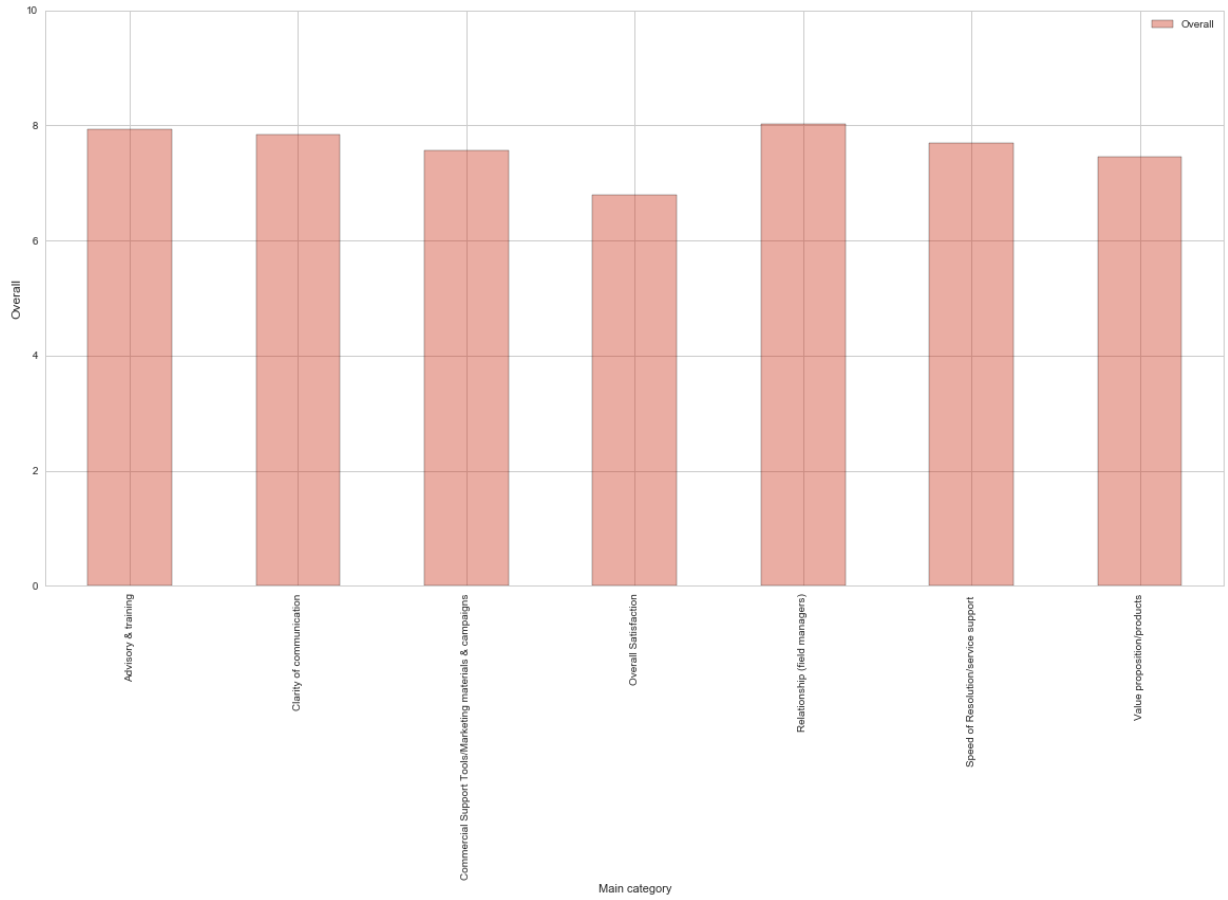


We observe that generally not much changes in the tendecies of the data between the pain points and the main categories, whereas the pain points give a more detailed view.

In [29]:
```
dfx = df2
dfx = dfx.groupby(['Main category'])['Overall'].mean()
%pylab inline
figsize(20,10)
dfx.plot.bar(color=(0.8, 0.2, 0.1, 0.4),  edgecolor='black', legend=True)
plt.ylabel("Overall", size = 12)
ylim(0,10)
```

```
Populating the interactive namespace from numpy and matplotlib
```

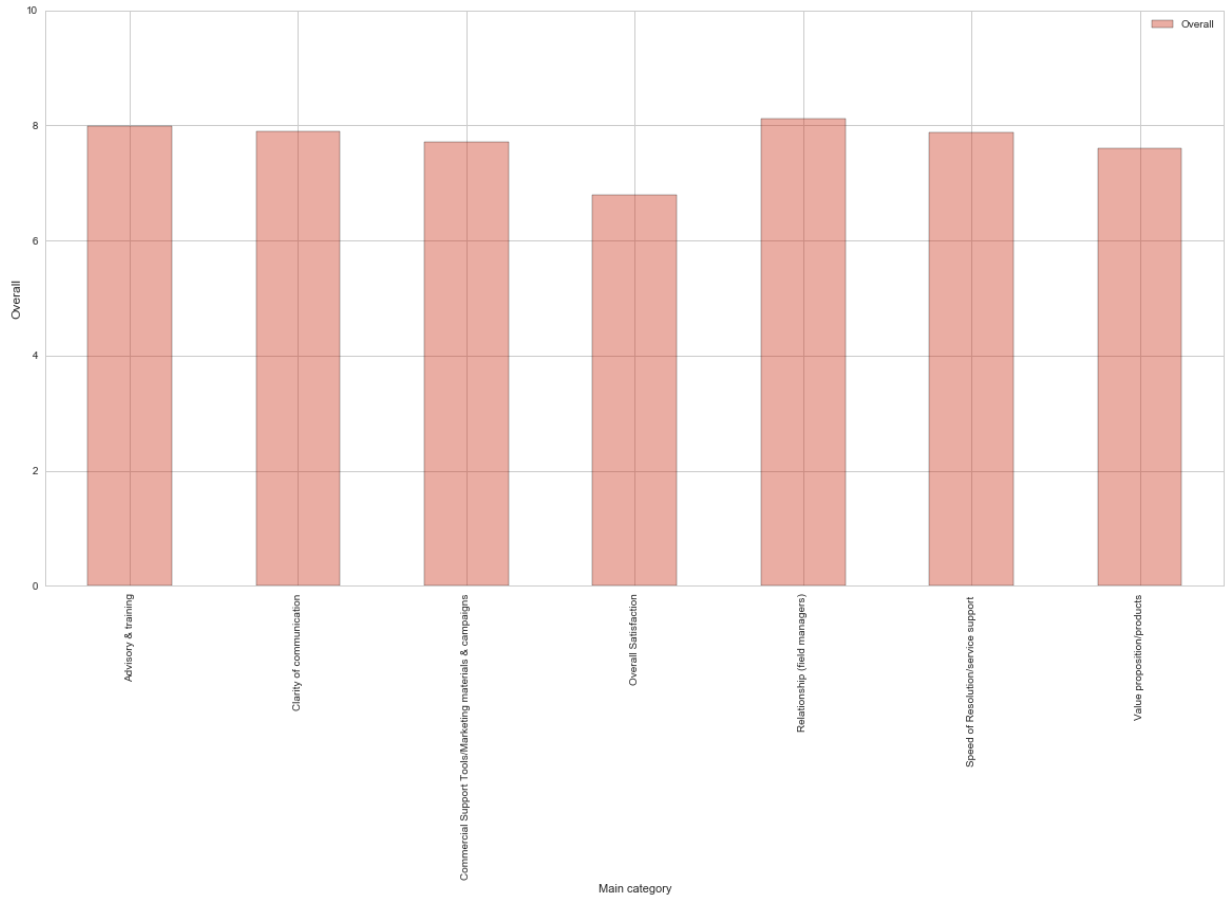
Out[29]:
```
(0, 10)
```



In [39]:
```
dfx = df2
dfx = dfx.groupby(['Main category'])['Overall'].median()
%pylab inline
figsize(20,10)
dfx.plot.bar(color=(0.8, 0.2, 0.1, 0.4),  edgecolor='black', legend=True)
plt.ylabel("Overall", size = 12)
ylim(0,10)
```

```
Populating the interactive namespace from numpy and matplotlib
```

Out[39]:
```
(0, 10)
```



Moreover, our interest now consists in analyzing and better understanding how exactly the values in each categorical variable are behaving and whether or not we may detect numerous outliers. Firstly, we used a GroupBy-function with a specific threshold (set to 8 in the following case). We set our threshold to a specific number and convert every value in the column to 0 if its the specific value of a driver was lower than this threshold set, whereas we set it to 1 if its value is higher than the threshold.

In [25]:
```
dfx = pd.read_csv('DriverAnalysisNT.csv')
#we convert now the values of every single driver to 0 and 1
dfx["Overall"] = dfx["Overall"].map(lambda s :1  if s >= 8 else 0)
dfxq = dfx.groupby(["Pain points", "Main category", "Overall"]).size()
dfxq
```

Out[25]:
```
Pain points                       Main category
Overall
Advisory & training               Advisory & training
0          21

1          21
Clarity of communication          Clarity of communication
0          26

1          21
Commercial Support Tools          Commercial Support Tools/Marketing materials &
campaigns  0          29

1          13
Marketing materials & campaigns   Commercial Support Tools/Marketing materials &
campaigns  0          12

1           6
Overall Satisfaction              Overall Satisfaction
0           1
Products                          Value proposition/products
0          53

1          27
Relationship (field managers)     Relationship (field managers)
0          23

1          31
Service support                   Speed of Resolution/service support
0          34

1          33
Speed of resolution               Speed of Resolution/service support
0          33

1          18
Value proposition                 Value proposition/products
0          62

1          19
dtype: int64
```

The preceding code is useful, however, it does not specify the percentages of the every single categorical variable, neither does it visualize them accordingly. As for our next step we now are interested in the percentage changes of every main category and pain point, whereas we are using multiple thresholds now in this case. To get clearer results, we now divide the analysis between Pain points and Main categories.

In [35]:
```
x = [6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5]
for i in x:
    dfx = pd.read_csv('DriverAnalysisNT.csv')
    dfx["Overall"] = dfx["Overall"].map(lambda s :1  if s >= i else 0)
    dfxq = dfx.groupby('Pain points')["Overall"].apply(lambda c: (c>0).sum()/len(
    print("The percentages of values of a Pain point above", i)
    print(dfxq)
    figsize(20,10)
```

```
The percentages of values of a Pain point above 6
Pain points
Advisory & training                1.000000
Clarity of communication           1.000000
Commercial Support Tools           1.000000
Marketing materials & campaigns    0.944444
Overall Satisfaction               1.000000
Products                           0.925000
Relationship (field managers)      0.962963
Service support                    0.970149
Speed of resolution                0.901961
Value proposition                  0.901235
Name: Overall, dtype: float64
The percentages of values of a Pain point above 6.5
Pain points
Advisory & training                0.952381
Clarity of communication           0.936170
Commercial Support Tools           0.952381
Marketing materials & campaigns    0.777778
Overall Satisfaction               1.000000
Products                           0.875000
Relationship (field managers)      0.907407
Service support                    0.940299
Speed of resolution                0.843137
Value proposition                  0.876543
Name: Overall, dtype: float64
The percentages of values of a Pain point above 7
Pain points
Advisory & training                0.857143
Clarity of communication           0.851064
Commercial Support Tools           0.690476
Marketing materials & campaigns    0.722222
Overall Satisfaction               0.000000
Products                           0.775000
Relationship (field managers)      0.907407
Service support                    0.850746
Speed of resolution                0.745098
Value proposition                  0.753086
Name: Overall, dtype: float64
The percentages of values of a Pain point above 7.5
Pain points
Advisory & training                0.738095
Clarity of communication           0.702128
Commercial Support Tools           0.642857
Marketing materials & campaigns    0.555556
Overall Satisfaction               0.000000
Products                           0.612500
```

```
Relationship (field managers)      0.814815
Service support                    0.656716
Speed of resolution                0.627451
Value proposition                  0.567901
Name: Overall, dtype: float64
The percentages of values of a Pain point above 8
Pain points
Advisory & training                0.500000
Clarity of communication           0.446809
Commercial Support Tools           0.309524
Marketing materials & campaigns    0.333333
Overall Satisfaction               0.000000
Products                           0.337500
Relationship (field managers)      0.574074
Service support                    0.492537
Speed of resolution                0.352941
Value proposition                  0.234568
Name: Overall, dtype: float64
The percentages of values of a Pain point above 8.5
Pain points
Advisory & training                0.309524
Clarity of communication           0.255319
Commercial Support Tools           0.119048
Marketing materials & campaigns    0.111111
Overall Satisfaction               0.000000
Products                           0.087500
Relationship (field managers)      0.351852
Service support                    0.194030
Speed of resolution                0.196078
Value proposition                  0.098765
Name: Overall, dtype: float64
The percentages of values of a Pain point above 9
Pain points
Advisory & training                0.119048
Clarity of communication           0.000000
Commercial Support Tools           0.000000
Marketing materials & campaigns    0.000000
Overall Satisfaction               0.000000
Products                           0.025000
Relationship (field managers)      0.055556
Service support                    0.029851
Speed of resolution                0.019608
Value proposition                  0.000000
Name: Overall, dtype: float64
The percentages of values of a Pain point above 9.5
Pain points
Advisory & training                0.000000
Clarity of communication           0.000000
Commercial Support Tools           0.000000
Marketing materials & campaigns    0.000000
Overall Satisfaction               0.000000
Products                           0.012500
Relationship (field managers)      0.018519
Service support                    0.000000
Speed of resolution                0.000000
Value proposition                  0.000000
Name: Overall, dtype: float64
```

We can detect a gradual decrease of values in percentage as the thresholds increase. The Overall Satisfaction pain point, however, contains only one value (6.787) and thus its percentage values falls drastically from 1.00 to 0.00 between the thresholds 6.5-7.

We now perform the same analysis for the main categories.

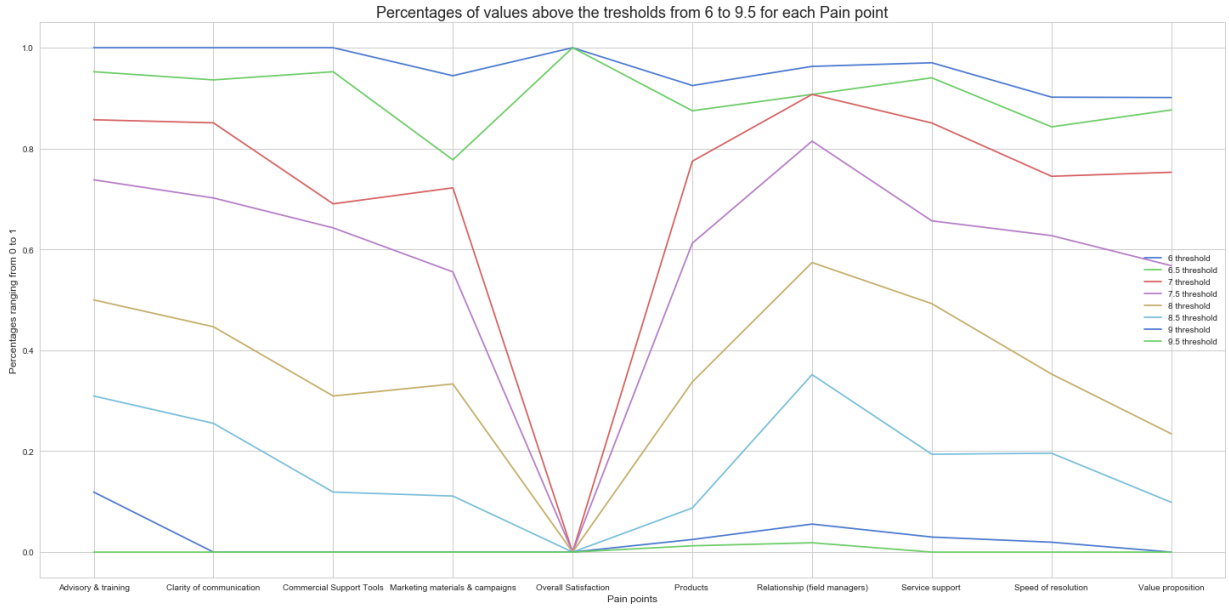
In [36]:
```
x = [6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5]
for i in x:
    dfx = pd.read_csv('DriverAnalysisNT.csv')
    dfx["Overall"] = dfx["Overall"].map(lambda s :1  if s >= i else 0)
    dfxq = dfx.groupby('Main category')["Overall"].apply(lambda c: (c>0).sum()/le
    print("The percentages of values of a Main category above", i)
    print(dfxq)
    figsize(20,10)
```

```
The percentages of values of a Main category above 6
Main category
Advisory & training                                          1.000000
Clarity of communication                                     1.000000
Commercial Support Tools/Marketing materials & campaigns     0.983333
Overall Satisfaction                                         1.000000
Relationship (field managers)                                0.962963
Speed of Resolution/service support                          0.940678
Value proposition/products                                   0.913043
Name: Overall, dtype: float64
The percentages of values of a Main category above 6.5
Main category
Advisory & training                                          0.952381
Clarity of communication                                     0.936170
Commercial Support Tools/Marketing materials & campaigns     0.900000
Overall Satisfaction                                         1.000000
Relationship (field managers)                                0.907407
Speed of Resolution/service support                          0.898305
Value proposition/products                                   0.875776
Name: Overall, dtype: float64
The percentages of values of a Main category above 7
Main category
Advisory & training                                          0.857143
Clarity of communication                                     0.851064
Commercial Support Tools/Marketing materials & campaigns     0.700000
Overall Satisfaction                                         0.000000
Relationship (field managers)                                0.907407
Speed of Resolution/service support                          0.805085
Value proposition/products                                   0.763975
Name: Overall, dtype: float64
The percentages of values of a Main category above 7.5
Main category
Advisory & training                                          0.738095
Clarity of communication                                     0.702128
Commercial Support Tools/Marketing materials & campaigns     0.616667
Overall Satisfaction                                         0.000000
Relationship (field managers)                                0.814815
Speed of Resolution/service support                          0.644068
Value proposition/products                                   0.590062
Name: Overall, dtype: float64
The percentages of values of a Main category above 8
Main category
Advisory & training                                          0.500000
Clarity of communication                                     0.446809
Commercial Support Tools/Marketing materials & campaigns     0.316667
Overall Satisfaction                                         0.000000
Relationship (field managers)                                0.574074
```

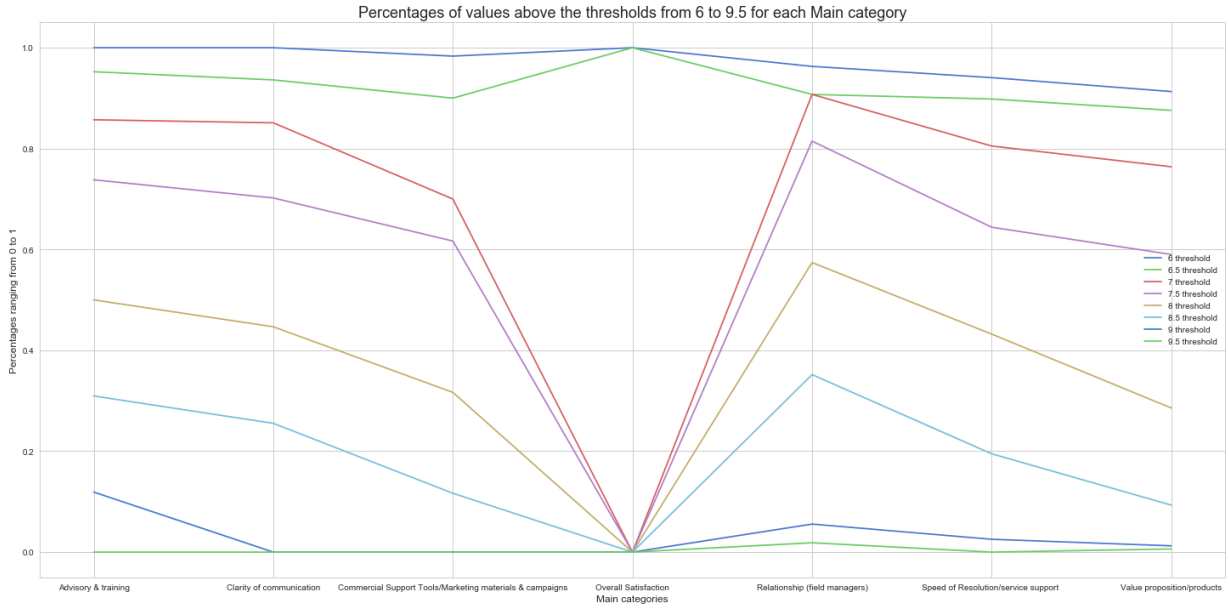
```
Speed of Resolution/service support                          0.432203
Value proposition/products                                   0.285714
Name: Overall, dtype: float64
The percentages of values of a Main category above 8.5
Main category
Advisory & training                                          0.309524
Clarity of communication                                     0.255319
Commercial Support Tools/Marketing materials & campaigns     0.116667
Overall Satisfaction                                         0.000000
Relationship (field managers)                                0.351852
Speed of Resolution/service support                          0.194915
Value proposition/products                                   0.093168
Name: Overall, dtype: float64
The percentages of values of a Main category above 9
Main category
Advisory & training                                          0.119048
Clarity of communication                                     0.000000
Commercial Support Tools/Marketing materials & campaigns     0.000000
Overall Satisfaction                                         0.000000
Relationship (field managers)                                0.055556
Speed of Resolution/service support                          0.025424
Value proposition/products                                   0.012422
Name: Overall, dtype: float64
The percentages of values of a Main category above 9.5
Main category
Advisory & training                                          0.000000
Clarity of communication                                     0.000000
Commercial Support Tools/Marketing materials & campaigns     0.000000
Overall Satisfaction                                         0.000000
Relationship (field managers)                                0.018519
Speed of Resolution/service support                          0.000000
Value proposition/products                                   0.006211
Name: Overall, dtype: float64
```

Our goal now is to visualize the previous findings into a multi-layered graph representing the percentages changes when introducing a higher treshhold.

In [164]:
```
thresholds = [6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5]
for i in thresholds:
    dfx = pd.read_csv('DriverAnalysisNT.csv')
    dfx["Overall"] = dfx["Overall"].map(lambda s :1  if s >= i else 0)
    dfxq = dfx.groupby('Pain points')["Overall"].apply(lambda c: (c>0).sum()/len(
    figsize(25,12)
    plt.title("Percentages of values above the tresholds from 6 to 9.5 for each P
    plt.ylabel("Percentages ranging from 0 to 1", size=12)
    plt.xlabel("Pain points", size=12)
    plot(dfxq, label='%s threshold' % i)
    plt.legend()
    plt.savefig("PainpointsPercentages2.png", dpi=200)
```



In [166]:
```
x = [6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5]
for i in x:
    dfx = pd.read_csv('DriverAnalysisNT.csv')
    dfx["Overall"] = dfx["Overall"].map(lambda s :1  if s >= i else 0)
    dfxq = dfx.groupby('Main category')["Overall"].apply(lambda c: (c>0).sum()/le
    figsize(25,12)
    plt.title("Percentages of values above the thresholds from 6 to 9.5 for each
    plt.ylabel("Percentages ranging from 0 to 1", size=12)
    plt.xlabel("Main categories", size=12)
    plot(dfxq, label='%s threshold' % i)
    plt.legend(loc="right")
    plt.savefig("MainCategoriesPercentages2.png", dpi=200)
```



Interestingly enough, the overall satisfaction always scores the lowest, pointing towards a negative attitude of suvery tzkers, in regard to the higher values of the company in each other main category/pain point.

We now convert the different categorical variable into numerical ones.

In [ ]:
```
df_ = df2
```

In [165]:
```
def multiplelines(x):

#MainCategories
    if x == 'Commercial Support Tools/Marketing materials & campaigns': #works
        x = 0
    elif x == 'Speed of Resolution/service support ':  #works
        x = 1
    elif x == 'Value proposition/products': #works
        x = 2
    elif x == 'Clarity of communication': #w
        x = 3
    elif x == 'Advisory & training':
        x = 4
    elif x == 'Relationship (field managers)':
        x = 5
    elif x == 'Overall Satisfaction':
        x = 9

    return x
```

In [166]:
```
def multiplelinesP(x):
#PainPoints
    if x == 'Marketing materials & campaigns': #works
        x = 0
    elif x == 'Speed of resolution': #works
        x = 1
    elif x == 'Service support ': #works
        x = 2
    elif x == 'Commercial Support Tools':  #works
        x = 3
    elif x == 'Value proposition': #works
        x = 4
    elif x == 'Products ': #works
        x = 5
    elif x == 'Clarity of communication':  #works
        x = 6
    elif x == 'Advisory & training': #works
        x = 7
    elif x == 'Relationship (field managers)': #works
        x = 8
    elif x == 'Overall Satisfaction':
        x = 9

    return x
```

In [167]:
```
df_['Main category'] = df_['Main category'].map(multiplelines)
df_['Pain points'] = df_['Pain points'].map(multiplelinesP)
df3 = df2
```

In [168]:
```
df_.head(50)
```

Out[168]:

	Unnamed: 0	Pain points	Main category	Jan	Feb	Mar	Apr	May	Jur
0	Overall Satisfaction	9	9	NaN	NaN	NaN	NaN	NaN	NaN
1	Ability to reach required service	1	1	5.214286	6.050000	6.047619	6.142857	NaN	6.16326
2	Ability to reach to regional sales team	1	1	NaN	NaN	NaN	NaN	NaN	NaN
3	Ability to resolve my issues	4	2	NaN	NaN	NaN	NaN	NaN	NaN
4	Accessibility to right person	1	1	NaN	NaN	NaN	NaN	NaN	NaN

In [79]:
```
df2.head(500)
```

Out[79]:

	Unnamed: 0	Pain points	Main category	Jan	Feb	Mar	Apr	May	
0	Overall Satisfaction	9	9	NaN	NaN	NaN	NaN	NaN	
1	Ability to reach required service	1	1	5.214286	6.050000	6.047619	6.142857	NaN	6.16
2	Ability to reach to regional sales team	1	1	NaN	NaN	NaN	NaN	NaN	
3	Ability to resolve my issues	4	2	NaN	NaN	NaN	NaN	NaN	
4	Accessibility to right person (Life)	1	1	NaN	NaN	NaN	NaN	NaN	
5	Accessibility to right person (Non	1	1	NaN	NaN	NaN	NaN	6.428571	6.06

We now delete assign these specific columns to their own variable.

In [66]:
```
Painpoints = df2['Pain points']
Maincategories = df2['Main category']
Surveys = df2['# Surveys']
Weight = df2['Weight']
Overall = df2['Overall']
```

In [16]:
```
columns = []
for i in df2['Unnamed: 0']:
    columns.append(i)
print(columns)
```

```
['Overall Satisfaction', 'Ability to reach required service', 'Ability to re
ach to regional sales team', 'Ability to resolve my issues', 'Accessibility
to right person (Life)', 'Accessibility to right person (Non Life)', 'Accide
nt', 'Accidents', 'Accounting Centres', 'Activity of sales support experts',
'Additional coverage', 'Additional coverage sat', 'Administration efficiency
of Relationship Managers', 'Administration efficiency of Sales Support Tea
m', 'App Calculations', 'App Ease', 'App Support', 'App Support Presentatio
n', 'Are you happy with the training opportunities that you receive from Gen
erali?', 'Availability (Account Management)', 'Availability of inspector',
'Availability of Partner Portal', 'Availability of your main contact', 'Avai
lability of your main contact', 'Availability of your main contact', 'Awaren
ess of news, campaigns, innovations and changes', 'Branch - Availability',
'Branch - Manager', 'Branch - Ops Execs', 'Branch - Support', 'Branch assign
ment', 'Business Opportunity Presentation (BOP)', 'Career opportunity', 'Cla
im Assistance', 'Claims', 'Claims - Clarity of process', 'Claims - Ease of c
ontact', 'Claims - Time to receive payment', 'Claims (Car repair shop)', 'Cl
aims (Invoices)', 'Claims Centre', 'Claims handling satisfaction', 'Claims l
iquidation', 'Claims Motor', 'Claims Non-Motor', 'Claims overall', 'Claims S
atisfaction', 'Clarity around claims handling process', 'Clarity of procedur
```

We now delete all not directly survey related columns, as we now analyze the specific values of the surveys.

In [45]:
```
del df2['Pain points']
del df2['Main category']
del df2['# Surveys']
del df2['Weight']
del df2['Overall']
del df2['Unnamed: 0']
```

In [51]:
```
df2.head()
```

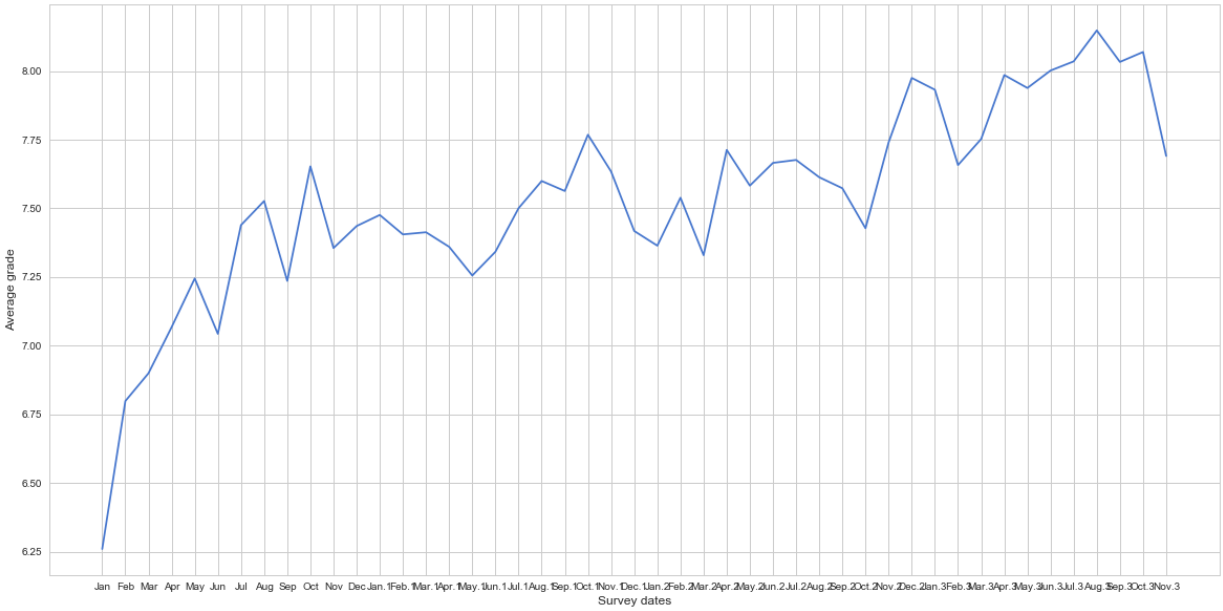
Out[51]:

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	6.2	6.250000	5.843750	6.210526	5
1	5.214286	6.05	6.047619	6.142857	NaN	6.163265	NaN	NaN	5.350877	8.000000	NaN	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	9.074627	NaN	9
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	8.194030	NaN	8
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Now we are interested in plotting the changes in values in general of all survey-votes received, based on their date and try to map out where this trend is currently going. We perform this in both the cases of the mean and the median.

In [46]:
```
y = df2.mean()
%pylab inline
figsize(20,10)
#ax = subplot(111)
plt.xlabel("Survey dates", size = 12)
plt.ylabel("Average grade", size = 12)
plt.plot(y)
plt.show()
```

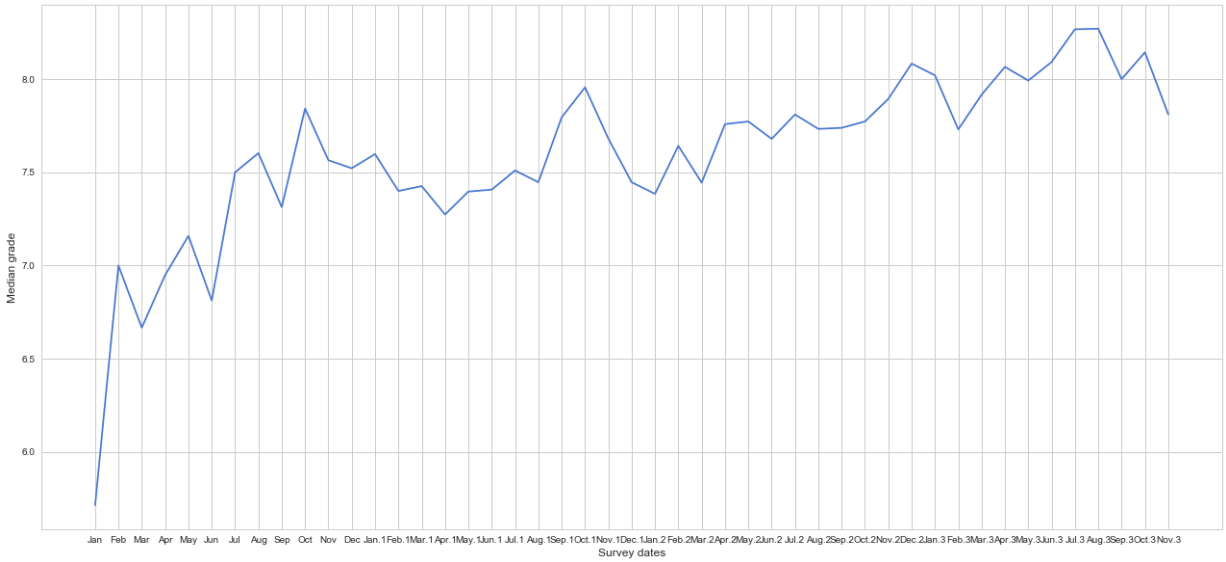
```
Populating the interactive namespace from numpy and matplotlib
```



We may note that there has been a distinct increase in values over the duration of these surveys. Notably, even though the average Overall satisfaction score is lower than 7, we can detect in the two graphs that the average and medians of all the single values have not been lower than this threshold since the July of the first year (40 months ago as of November 2018).

In [47]:
```
y = df2.median()
%pylab inline
figsize(22,10)
#ax = subplot(111)
plt.xlabel("Survey dates", size = 12)
plt.ylabel("Median grade", size = 12)
plt.plot(y)
plt.show()
```

```
Populating the interactive namespace from numpy and matplotlib
```



Only really small differences may be detected between the mean and the median, whereas the increasing trend is clear.

In conclusion, over the period of time the values have, in general, been nearly strictly increasing, whereas our two categorical columns have given us the ability to inspect more thoroughly the patterns of the data.