projekt kivitendo - Fehler #277

Kontoauszug verbuchen. Vorschlagsliste ignoriert SEPA-Überweisungen

23.07.2017 18:17 - Jan Büren

Status:	Erledigt	**Beginn:**	23.07.2017
Priorität:	Normal	**Abgabedatum:**	18.08.2017
Zugewiesen an:	Martin Helmling	**% erledigt:**	100%
Kategorie:		**Geschätzter Aufwand:**	12.00 Stunden
Zielversion:	3.5.1	**Aufgewendete Zeit:**	0.00 Stunde

Beschreibung

Ist seit:
88d162cc5e8e8d3810059d918a1eef6a2b205984

defekt.

Anbei ein Screenshot inkl. korrekter Punktzahl mit Programmstand von Commit 8b14060

Ab Commit 88d162c bekommen SEPA-Überweisungen 0 Punkte.

Der Commit verbessert einen SEPA-Erkennungsfall (SEPA-Sammelüberweisungen) und zerstört gleichzeitig das alte Verfahren.

Ich bin jetzt zu 50% im Code drin, einen Hotfix der es schon etwas besser macht hab ich auch.

Ich würde mir aber folgende Überarbeitung wünschen:

a) Testfall für sepa_export_items OHNE Sammelüberweisung schreiben
b) Die sehr exakte und gut kommentierte Anweisung in get_agreement_with_bank_transactions (Überprüfung auf SEPA-Items) bitte mehr respektieren und wiederherstellen
c) Punktevergabe für den Fall SEPA-Sammelüberweisung durch einen besseren Algorithmus auch in der obigen Funktion abbilden und aus dem Controller rausnehmen oder eigene Funktion dafür schreiben.

Danke!

Zugehörige Revisionen

Revision 503fabbf - 10.08.2017 14:30 - Martin Helmling martin.helmling@octosoft.eu

BankTransaction: Überarbeitung von "Kontoauszug verbuchen" , SEPA-Export wieder integriert

Die Punktebewertung findet wieder ausschließlich in "get_agreement_with_bank_transactions" statt,
auch die SEPA-Sammelüberweisung. Diese bekommt dor extra Punkte, da ggf. für bestimmte Rechnungen negative Punkte entstehen.
Auch gibt es dort keine Remote Banknummer etc.

Die Testdatei t/bank/bank_transactions.t wurde um zwei Tests erweitert,
1. ein Test der das Verbuchen ohne SEPA-Export macht,
2. ein Test mit SEPA-Export

fixt #277

Historie

#1 - 02.08.2017 09:08 - Martin Helmling

- Abgabedatum wurde auf 18.08.2017 gesetzt

- Zugewiesen an wurde auf Martin Helmling gesetzt

- Zielversion wurde auf 3.5.1 gesetzt

- Geschätzter Aufwand wurde auf 12.00 h gesetzt

Hier gibt es auch noch andere Ungereimtheiten.

Werde mich nächste Woche darum kümmern können.

#2 - 02.08.2017 10:46 - Jan Büren

Ok, danke.

Hier, einige Ideen, die mir an Ungereimtheiten aufgefallen sind:

1.)

Die Manipulation an dem Array @all_sepa_invoices finden nur lokal statt und werden nie an der Oberfläche, bzw. später ausgewertet.
Ob ich hier agreement +5 gebe oder Kuh mit Milch zuweisen, spielt keine Rolle:

```
            if($bt->{remote_account_number} eq $iban && abs(abs($open_invoice->amount) - abs($bt->amount)) < 0
.01 ) {
              ($open_invoice->{agreement}, $open_invoice->{rule_matches}) = $bt->get_agreement_with_invoice($o
pen_invoice);
              $open_invoice->{agreement} += 5;
              $open_invoice->{rule_matches} .= 'sepa_export_item(5) ';
              # it really doesn't matter, what we do here
+             $open_invoice->{kuh}        = 'Milch und Muh!';
```

Wenn man da weiter schaut, liegt eine Ursache darin, dass immer gesamte Objekte innerhalb von Arrays verdoppelt und verdreifacht werden:
Ursprung:

```
  my @all_open_invoices;
```

Neue Datenkollektionen Kopien des Ursprungs:

```
  my @all_sepa_invoices;
  my @all_non_sepa_invoices;
```

An der Stelle steig ich aus und würde bei Sven oder Moritz nachfragen, was hier eine schlauere Implementierung ist (erste Idee: array refs oder nur mit ids arbeiten)

2.)

Geoff hat sich mit get_agreement_with_bt liebevoll Mühe gemacht, einen Anwendervorteil konfigurierbar zu gestalten.
Diese Mühe wird hiermit ignoriert und wir haben beim Umbau guten Code mit schlechterem Code ersetzt:

Vorher:

```
  my %points = (
    cust_vend_name_in_purpose   => 1,
    cust_vend_number_in_purpose => 1,
    datebonus0                  => 3,
    datebonus14                 => 2,
    datebonus35                 => 1,
     (...)
    wrong_sign                  => -1,
    sepa_export_item            => 5,
  );
```

Nachher:

```
-     sepa_export_item            => 5,

# Werte die vorher zentral in einem logischen Hash gepflegt wurden,
# besser hartkodieren und in eine andere Programmfunktion einbauen
+    $open_invoice->{agreement} += 5;
+              $open_invoice->{rule_matches} .= 'sepa_export_item(5) ';
```

Nicht falsch verstehen, in einem drängenden Kundenprojekt hätt ich das
genauso reingehackt ohne nach links und rechts zu schauen, um die Anforderung
vom Tisch zu haben!

Für den Standard erwarte ich aber hier, ehrlich gesagt, ein höheres Niveau als
für ein Kundenprojekt.

Der BankTransaction Controller Code ist weder gut noch schlecht, dass macht ihn
schwierig für Umbauarbeiten.
Einem Testfall vorab, der die Umbaumaßnahmen alte Erwartungen neue Erwartungen kontrolliert hätte,
wäre gut gewesen.

3.)

Mir ist die Begründung dieser Auskommentierung nicht klar:

```
# Rollback Fehler nicht weiterreichen
# die if $error;
```

Ich hab probiert den Fall zu provozieren, bekomme aber keine Fehlermeldung mehr bei einer negativen Kreditorenbuchung.
Der Verdacht steigt in mir hoch, dass das Auskommentieren nur das Sympton entfernt hat und der wahre Fix später im Projektverlauf dazukam.
Ich würde lieber das "die" wieder reinkommentieren und bei der nächsten Fall, vorher nochmal das Datenmodell gegendenken.

Soweit, dass was ich vorgedacht hab.

#3 - 10.08.2017 14:29 - Anonym

- Status wurde von Neu zu Gelöst geändert

- % erledigt wurde von 0 zu 100 geändert

Status geändert durch Changeset kivitendo-erp|commit:503fabbf4b2c77b2aab2dc1940f1eec0842cd490.

#4 - 13.08.2017 14:33 - Jan Büren

- Status wurde von Gelöst zu Neu geändert

- % erledigt wurde von 100 zu 50 geändert

Hi,
lustigerweise ist der ursprüngliche Fehler noch nicht ganz behoben, der ist aber dem schlechten Design geschuldet:

Reproduzierbar wie folgt:

a) DB-Bestand mit SEPA-Überweisungen einspielen

Import-Dateien wieder neu verwenden (in DB):

```
update ap set paid=0;
delete from  acc_trans where chart_link ilike '%paid%';
update bank_transactions set cleared='f',invoice_amount=0;
```

b) Abstürze provozieren:

SL/DB/BankTransaction.pm

```
sub get_agreement_with_bank {

      if ( abs(abs($self->amount) - abs($sei->amount)) < 0.01 ) {
+        use Carp;
+        croak ("Wir sind unsichere Codezeilen. Wir hängen von Randbedingungen in anderen Routinen ab. Vertrau
t uns nicht! Schlagt nicht diesen Weg ein! Steinschlag und Minenfelder drohen hier. Vertraut auf Euren guten a
lten Boyscout Geoffrey in der Vorgängerversion, nur DER kennt den wahren SEPA-Export weg in die Freiheit!");
        $agreement    += $points{sepa_export_item};
        $rule_matches .= 'sepa_export_item(' . $points{'sepa_export_item'} . ') ';
      }
    } else {
```

So, wir entfernen die if-Bedingung in Controller/BankTransaction.pm

```
+       #if ( $_->ap_id == $open_invoice->id ||  $_->ar_id == $open_invoice->id ) {
         my $factor                   = ($_->ar_id == $open_invoice->id ? 1 : -1);
         #$main::lxdebug->message(LXDebug->DEBUG2(),"sepa_exitem=".$_->id." for invoice ".$open_invoice->id." f
actor=".$factor);
         $open_invoice->{realamount}  = $::form->format_amount(\%::myconfig,$open_invoice->amount*$factor,2);
         push @{$open_invoice->{sepa_export_item}}, $_;
         $open_invoice->{skonto_type} = $_->payment_type;
         $sepa_exports{$_->sepa_export_id} ||= { count => 0, is_ar => 0, amount => 0, proposed => 0, invoices =
> [], item => $_ };
         $sepa_exports{$_->sepa_export_id}->{count}++;
         $sepa_exports{$_->sepa_export_id}->{is_ar}++ if  $_->ar_id == $open_invoice->id;
         $sepa_exports{$_->sepa_export_id}->{amount} += $_->amount * $factor;
         push @{ $sepa_exports{$_->sepa_export_id}->{invoices} }, $open_invoice;
+       #}
```

So, jetzt schau ich mir die Änderungen im Detail an:

```
   # if there is exactly one non-executed sepa_export_item for the invoice
-  if ( my $seis = $invoice->find_sepa_export_items({ executed => 0 }) ) {
-    if (scalar @$seis == 1) {
+  if ( my $seis = $invoice->{sepa_export_item} ) {
+      if (scalar @$seis == 1) {
       my $sei = $seis->[0];

-      # test for amount and id matching only, sepa transfer date and bank
-      # transaction date needn't match
-      my $arap = $invoice->is_sales ? 'ar' : 'ap';
-
-      if (abs($self->amount) == ($sei->amount) && $invoice->id == $sei->arap_id) {
+      if ( abs(abs($self->amount) - abs($sei->amount)) < 0.01 ) {
         $agreement    += $points{sepa_export_item};
         $rule_matches .= 'sepa_export_item(' . $points{'sepa_export_item'} . ') ';
       }
```

Es sind 7 Zeilen Geoffrey Code gelöscht und drei Zeilen Martin Code hinzugefügt.
In den drei Zeilen befindet sich ein Kosmetik-Fehler.
Ferner verstehe ich nicht die Verbesserung von:

```
-      if (abs($self->amount) == ($sei->amount) && $invoice->id == $sei->arap_id) {
+      if ( abs(abs($self->amount) - abs($sei->amount)) < 0.01 ) {
```

Kannst Du die erklären? Geoffs Code fühlt sich wesentlich stabiler und exakter an.

Und an dieser Stelle, bist Du instabiler als der vorherige Code:

```
+  if ( my $seis = $invoice->{sepa_export_item} ) {
+      if (scalar @$seis == 1) {
```

Gruß

#5 - 13.08.2017 21:29 - Jan Büren

- Status wurde von Neu zu In Bearbeitung geändert

- % erledigt wurde von 50 zu 80 geändert

Ich hab mir jetzt mal die Mühe gemacht, vorher mit nachher genauer zu analysieren und dabei festgestellt, dass das Zurücksetzen des DB-Bestands noch ein

```
update sepa_export_items set executed='f';
```

Benötigt.
Damit finde ich dann auch wieder meine Stichprobe korrekt und ich hab mal beide Implementierung (Vorher/Nachher) (Geoff/Martin) parallel laufen lassen,
um die Ergebnisse zu vergleichen.

In meinem DB-Bestand befinden sich 59 SEPA-Export Items.

Hier das Endergebnis des heutigen Spiels:

Martin vs Geoff 11 : 23

```
2017-08-13 21:08:34.890 1934 [1934] : geoff treffer bei7102442 1071.14000 -1071.14
2017-08-13 21:08:41.446 1934 [1934] : geoff treffer bei67/67-2901-05672241-8 1. Beitrag 391.04000 -391.04
2017-08-13 21:08:48.126 1934 [1934] : geoff treffer beiD5/10780 2908.80000 -2908.8
2017-08-13 21:08:54.906 1934 [1934] : geoff treffer bei038151 190.40000 -190.4
2017-08-13 21:09:01.609 1934 [1934] : geoff treffer beiD5/10780 5292.00000 -5292
2017-08-13 21:09:41.489 1934 [1934] : geoff treffer bei1100600103 14160.00000 -14160
2017-08-13 21:09:41.489 1934 [1934] : martin treffer bei1100600103 14160.00000 -14160
2017-08-13 21:09:48.910 1934 [1934] : geoff treffer beiD5/10804 6196.50000 -6196.5
2017-08-13 21:09:48.910 1934 [1934] : martin treffer beiD5/10804 6196.50000 -6196.5
2017-08-13 21:10:02.400 1934 [1934] : geoff treffer bei1100600103 206.00000 -206
2017-08-13 21:10:02.400 1934 [1934] : martin treffer bei1100600103 206.00000 -206
2017-08-13 21:10:09.140 1934 [1934] : geoff treffer beiV6/474 964.80000 -964.8
2017-08-13 21:10:09.140 1934 [1934] : martin treffer beiV6/474 964.80000 -964.8
2017-08-13 21:10:15.437 1934 [1934] : geoff treffer beiD5/10878 3628.80000 -3628.8
2017-08-13 21:10:15.437 1934 [1934] : martin treffer beiD5/10878 3628.80000 -3628.8
2017-08-13 21:10:22.711 1934 [1934] : geoff treffer bei94026933 12200.00000 -12200
2017-08-13 21:10:22.711 1934 [1934] : martin treffer bei94026933 12200.00000 -12200
2017-08-13 21:10:29.376 1934 [1934] : geoff treffer bei275/017 790.00000 -790
```

```
2017-08-13 21:10:29.376 1934 [1934] : martin treffer bei275/017 790.00000 -790
2017-08-13 21:10:36.045 1934 [1934] : geoff treffer bei233 307 875 // 1/2017 34.98000 -34.98
2017-08-13 21:10:36.045 1934 [1934] : martin treffer bei233 307 875 // 1/2017 34.98000 -34.98
2017-08-13 21:10:42.274 1934 [1934] : geoff treffer beiD5/10788 21945.00000 -21945
2017-08-13 21:10:42.274 1934 [1934] : martin treffer beiD5/10788 21945.00000 -21945
2017-08-13 21:10:49.540 1934 [1934] : geoff treffer beiD5/10788 39501.00000 -39501
2017-08-13 21:10:49.540 1934 [1934] : martin treffer beiD5/10788 39501.00000 -39501
2017-08-13 21:10:56.151 1934 [1934] : geoff treffer beiFNA170290 1350.00000 -1350
2017-08-13 21:10:56.151 1934 [1934] : martin treffer beiFNA170290 1350.00000 -1350
2017-08-13 21:11:29.343 1934 [1934] : geoff treffer bei11301/2017/10392/2017 333.20000 -333.2
2017-08-13 21:11:35.500 1934 [1934] : geoff treffer bei1706-15-124 456.00000 -456
2017-08-13 21:11:42.093 1934 [1934] : geoff treffer bei1706-15-124 33.90000 -33.9
2017-08-13 21:12:15.794 1934 [1934] : geoff treffer bei1706-07-117 11060.00000 -11060
2017-08-13 21:12:21.974 1934 [1934] : geoff treffer bei1705-26-112 1130.00000 -1130
2017-08-13 21:12:48.991 1934 [1934] : geoff treffer bei3706/VV/SE 50.00000 -50
2017-08-13 21:12:55.569 1934 [1934] : geoff treffer bei11301/2017/10391/2017 1053.57000 -1053.57
```

Klar, ich kann jetzt nichts über die Qualität der Treffer sagen.

Aber die Tendenz, dass der schlechter lesbare Code, der zusätzlich Bedingungen aus dem Controller benötigt und scheinbar schlechtere Ergebnisse liefert macht das Gesamtbild gerade nicht gut.

Ich lasse mich gerne von einem aussagekräftigeren Test/Testprotokoll eines besseren Belehren!

Hier die beiden Code-Schnipsel, die gegeneinander bei mir angetreten sind:

```
  if ( my $seis = $invoice->find_sepa_export_items({ executed => 0 }) ) {
    if (scalar @$seis == 1) {
      my $sei = $seis->[0];

      # test for amount and id matching only, sepa transfer date and bank
      # transaction date needn't match
      my $arap = $invoice->is_sales ? 'ar' : 'ap';

      if (abs($self->amount) == ($sei->amount) && $invoice->id == $sei->arap_id) {
        $agreement    += $points{sepa_export_item};
        $rule_matches .= 'sepa_export_item(' . $points{'sepa_export_item'} . ') ';
        $geoff++;
        $main::lxdebug->message(0, 'geoff treffer bei' . $invoice->invnumber . ' ' . $invoice->amount . ' ' .
$self->amount);
      }
    }
  }

  # if there is exactly one non-executed sepa_export_item for the invoice
  # if ( my $seis = $invoice->sepa_export_items ) {
  if ( my $seis = $invoice->{sepa_export_item} ) {
      if (scalar @$seis == 1) {
      my $sei = $seis->[0];
      #croak "hier b" . $sei->amount ." " . scalar @$seis if $invoice->id eq '9638' && $self->id == 6;
      #croak ("wtf $self->amount mit $sei->amount") if $invoice->id eq '9638' && $self->id == 6;
      if ( abs(abs($self->amount) - abs($sei->amount)) < 0.01 ) {
       # croak ("Wir sind tote Codezeilen. Wir werden nur von einem Test ausgeführt" . Dumper($invoice->{sepa_
export_item}));
        $agreement    += $points{sepa_export_item};
        $rule_matches .= 'sepa_export_item(' . $points{'sepa_export_item'} . ') ';
        $martin++;
        $main::lxdebug->message(0, 'martin treffer bei' . $invoice->invnumber . ' ' . $invoice->amount . ' ' .
 $self->amount);
      }
    }
```

#6 - 08.11.2017 11:00 - Jan Büren

- Status wurde von In Bearbeitung zu Erledigt geändert

- % erledigt wurde von 80 zu 100 geändert

Works for me

Dateien

sepa-export-punktzahl-io-kontoauszug-verbuchen.png	12,2 KB	23.07.2017	Jan Büren