

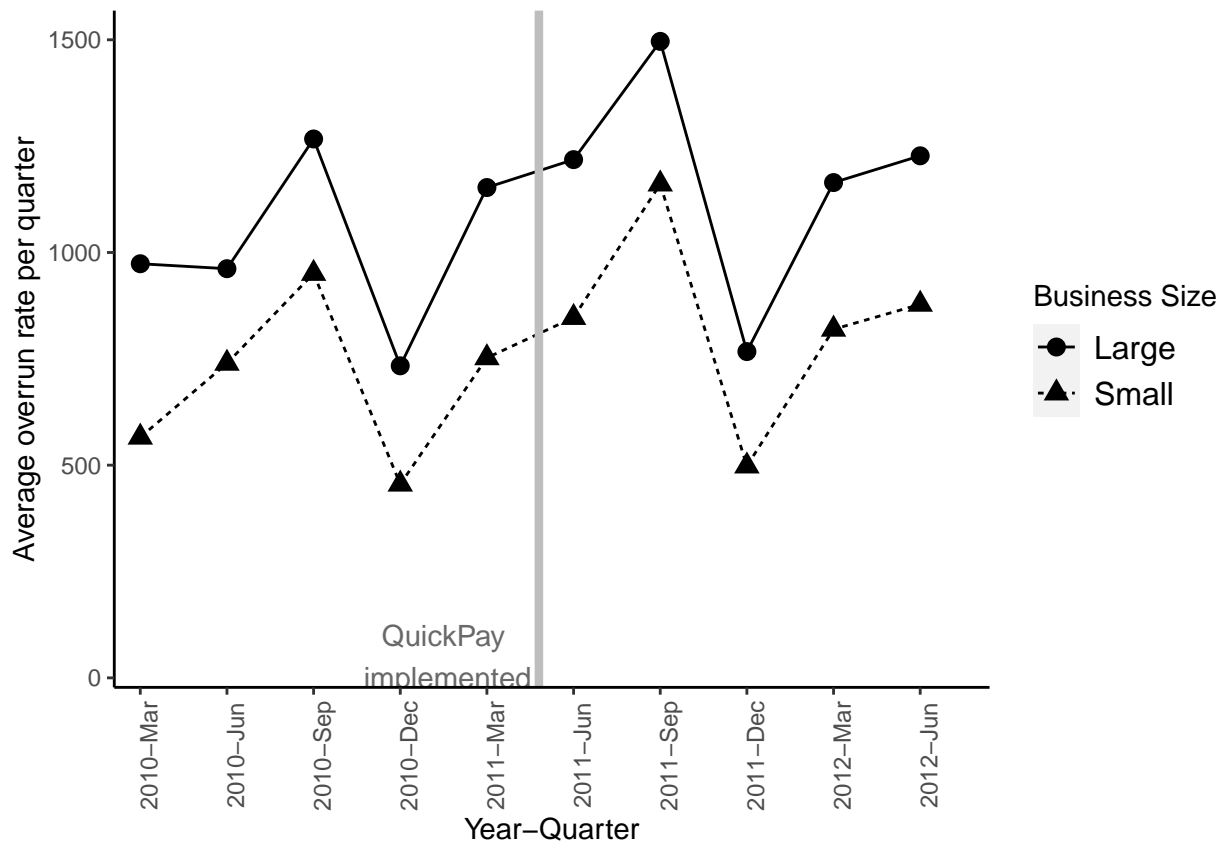
Budget Overruns: First Implementation of QuickPay (2009-2012)

Jan 17, 2022

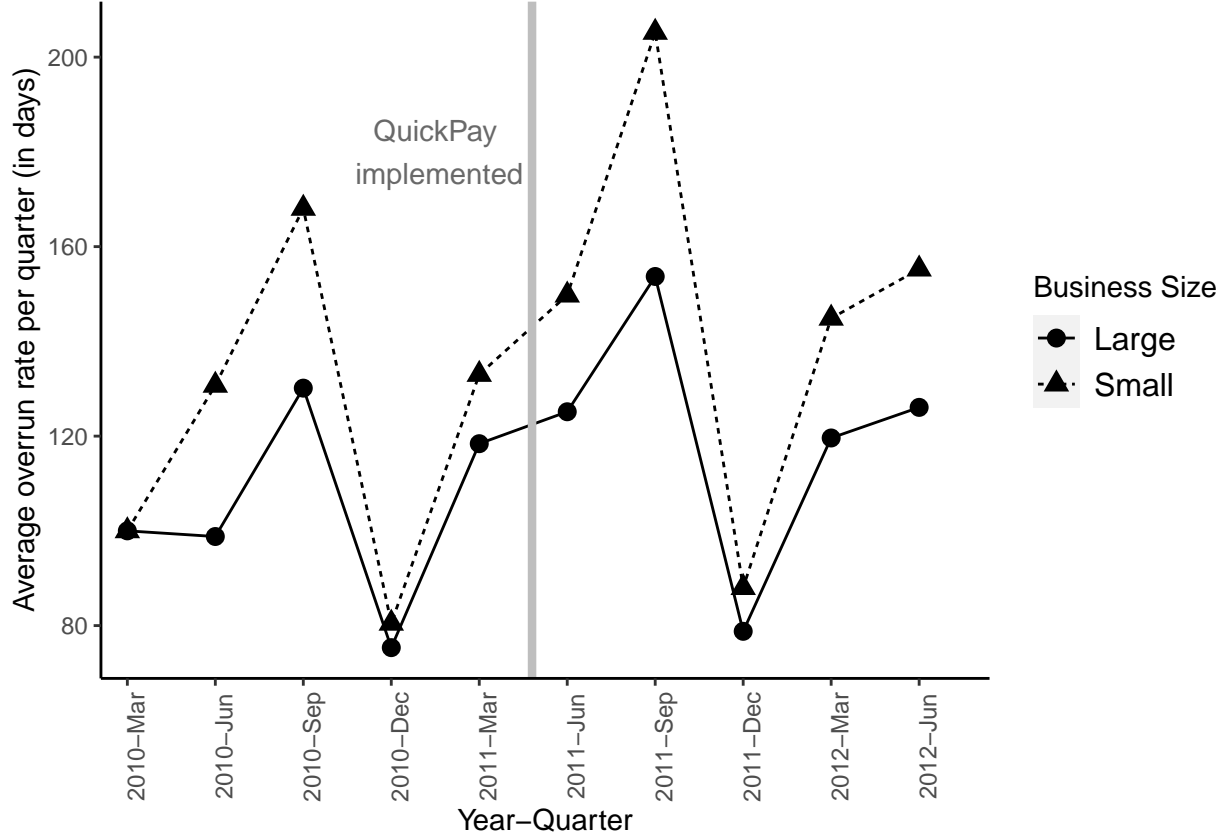
1 Note

- Sample restricted to projects for which start dates matches the one in API
 - This is done by using first reported “action_date” and “date_signed”
- Below is the definition of **base_and_all_options_value** from the data dictionary:
 - The change (from this transaction only) to the potential contract value (i.e., the base contract and any exercised or unexercised options).
- This means that every observation in raw data shows incremental change from previous budget. So some of the values can be zero.
- We, therefore, need to calculate the new budget at each point in time (by adding all previous values). We did this in the resampling step, but mentioning here for reference.
- This is different from calculation of delays, where **period_of_performance_current_end_date** indicated the new deadline of the project.

2 Budget Overrun over Time



2.1 Normalized Overrun



3 Notation

- Project i , Year-Quarter t
- X_i denotes project level controls: initial duration, initial budget, number of offers received
- $\mu_t, \theta_{firm}, \lambda_{task}$: Year-Quarter, Firm, and Product/Service code Fixed effects
- All continuous variables are winsorized at the 5% level

$$Treat_i = \begin{cases} 1, & \text{if project } i \text{ is a small business} \\ 0, & \text{otherwise} \end{cases}$$

$$Post_t = \begin{cases} 1, & \text{if year-quarter } t > \text{April 27, 2011} \\ 0, & \text{otherwise} \end{cases}$$

4 Baseline Regressions

$$Overrun_{it} = \alpha + \beta_0 Treat_i + \beta_1 Post_t + \beta_2 (Treat_i \times Post_t) + \epsilon_{it}$$

$$\begin{aligned} Overrun_{it} = & \alpha + \beta_0 Treat_i + \beta_1 Post_t + \beta_2 (Treat_i \times Post_t) \\ & + X_i + (Post_t \times X_i) + \mu_t + \theta_{firm} + \lambda_{task} + \epsilon_{it} \end{aligned}$$

Table 1: Quickpay 2009-2011

	<i>Overrun_{it}</i> (in days)				
	(1)	(2)	(3)	(4)	(5)
<i>Treat_i</i>	−329.68*** (25.53)	−81.18*** (25.95)	−75.66*** (25.93)	−42.03 (26.51)	−24.14 (26.63)
<i>Post_t</i>	143.11*** (23.43)	−299.19*** (39.87)			
<i>Treat_i × Post_t</i>	−3.92 (29.71)	21.68 (30.95)	18.52 (30.93)	19.51 (30.48)	19.29 (30.52)
Constant	1,014.64*** (20.39)	820.03*** (32.46)			
Duration, Budget, Bids	No	Yes	Yes	Yes	Yes
<i>Post_t × (Duration, Budget, Bids)</i>	No	Yes	Yes	Yes	Yes
Project Age Tercile	No	Yes	Yes	Yes	Yes
Year-Quarter Fixed Effects	No	No	Yes	Yes	Yes
Task Fixed Effects	No	No	No	Yes	Yes
Industry fixed effects	No	No	No	No	Yes
Observations	287,530	263,488	263,488	263,488	263,488
R ²	0.003	0.06	0.07	0.12	0.12
Adjusted R ²	0.003	0.06	0.07	0.11	0.11

Note:

*p<0.1; **p<0.05; ***p<0.01

Each observation is a project-quarter.

SEs are robust and clustered at the project level.

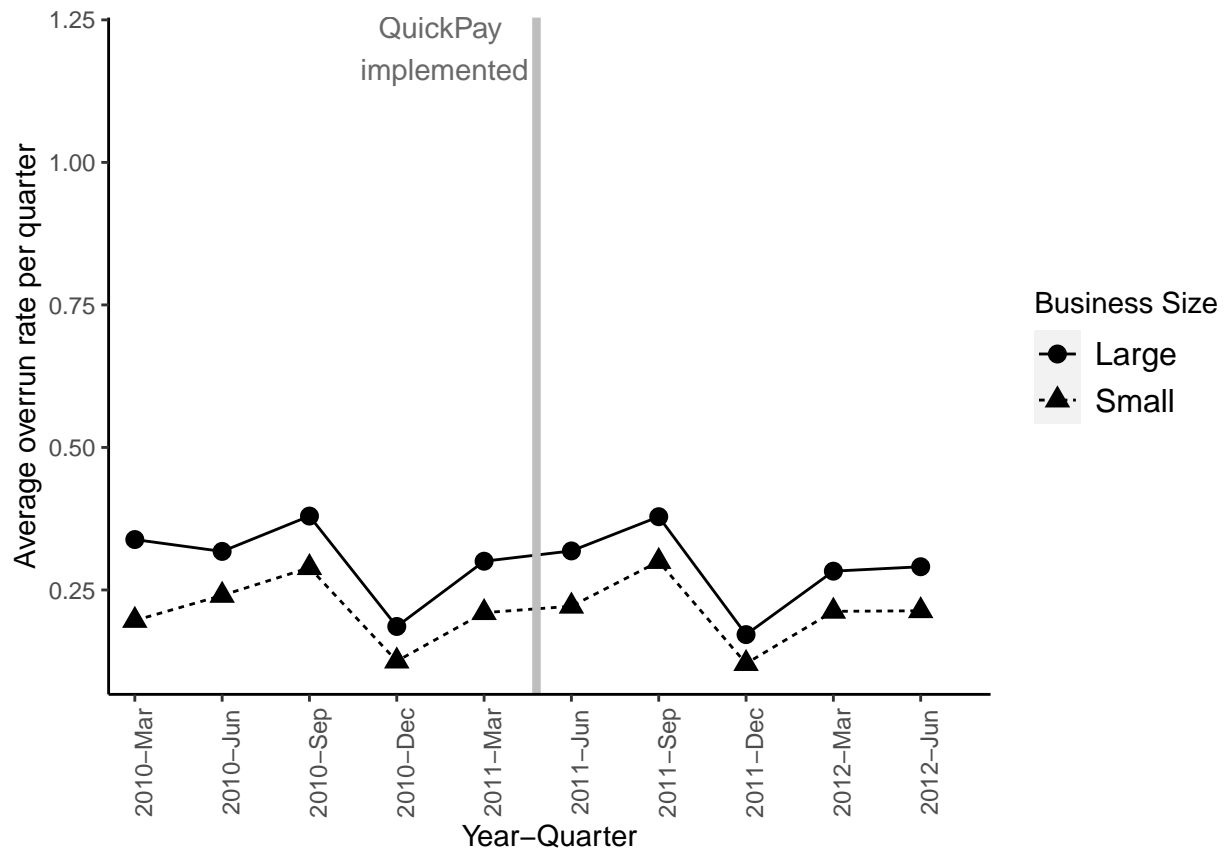
5 Percentage Overrun

$$PercentOverrun_{it} = \beta_0 + \beta_1 Treat_i + \beta_2 Post_t + \beta_3 (Treat_i \times Post_t) + e_{it}$$

$$\begin{aligned}
 PercentOverrun_{it} = & \alpha + \beta_0 Treat_i + \beta_1 Post_t + \beta_2 (Treat_i \times Post_t) \\
 & + X_i + (Post_t \times X_i) + \mu_t + \theta_{firm} + \lambda_{task} + \epsilon_{it}
 \end{aligned}$$

5.1 Percentage Overrun over time

- Sample restricted to projects with modification zero when they first appeared in our sample.
- $PercentOverrun_{it} = 100 \times Overrun_{it} / Budget_{i,t-1}$



5.1.1 Normalized Overrun

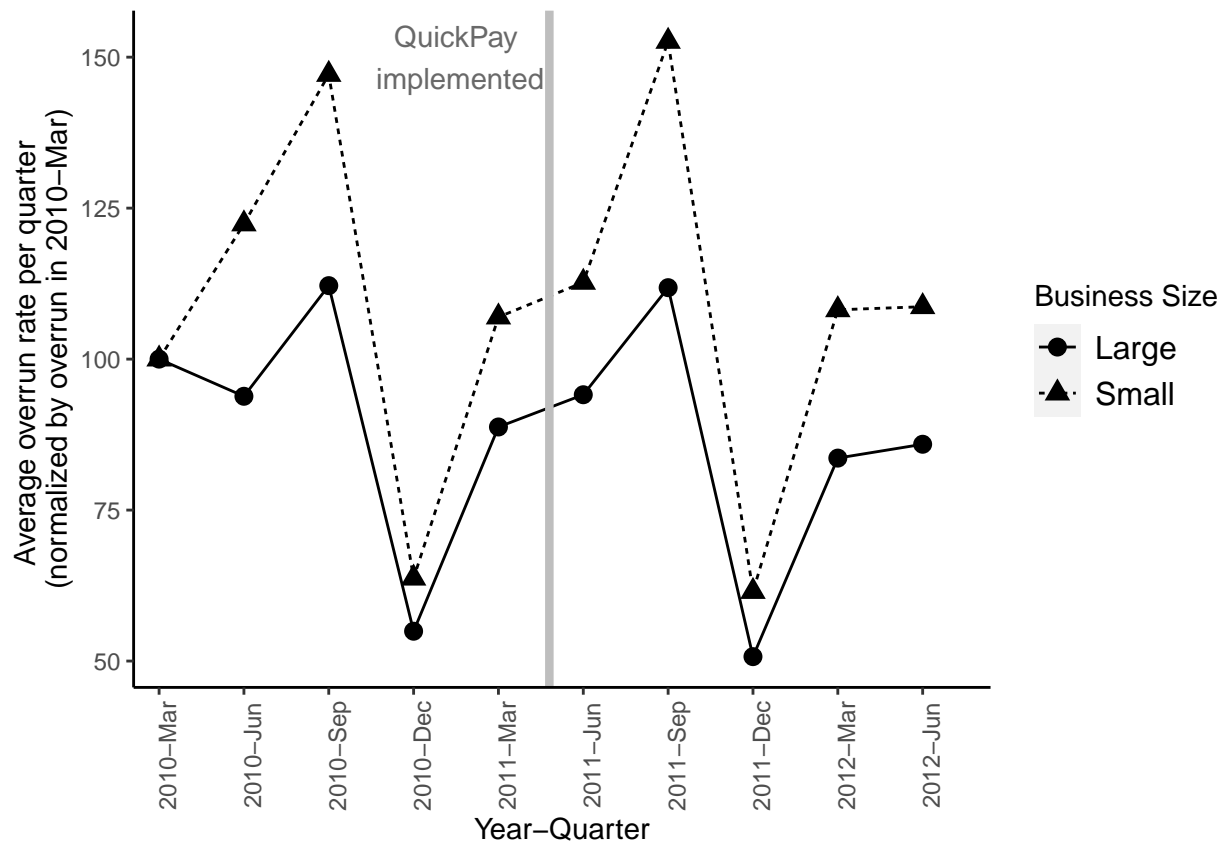


Table 2: Effect of QuickPay on project overrun rates

	<i>PercentOverrun_{it}</i>				
	(1)	(2)	(3)	(4)	(5)
<i>Treat_i</i>	-0.09*** (0.01)	-0.05*** (0.01)	-0.05*** (0.01)	-0.01 (0.01)	-0.001 (0.01)
<i>Post_t</i>	-0.01 (0.01)	-0.10*** (0.01)			
<i>Treat_i × Post_t</i>	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.0002 (0.01)	0.001 (0.01)
Constant	0.29*** (0.01)	0.43*** (0.01)			
Duration, Budget, Bids	No	Yes	Yes	Yes	Yes
<i>Post_t × (Duration, Budget, Bids)</i>	No	Yes	Yes	Yes	Yes
Project age	No	Yes	Yes	Yes	Yes
Year-Quarter fixed effects	No	No	Yes	Yes	Yes
Task fixed effects	No	No	No	Yes	Yes
Industry fixed effects	No	No	No	No	Yes
Observations	279,512	258,150	258,150	258,150	258,150
R ²	0.002	0.03	0.04	0.10	0.11
Adjusted R ²	0.002	0.03	0.04	0.10	0.10

Note:

*p<0.1; **p<0.05; ***p<0.01

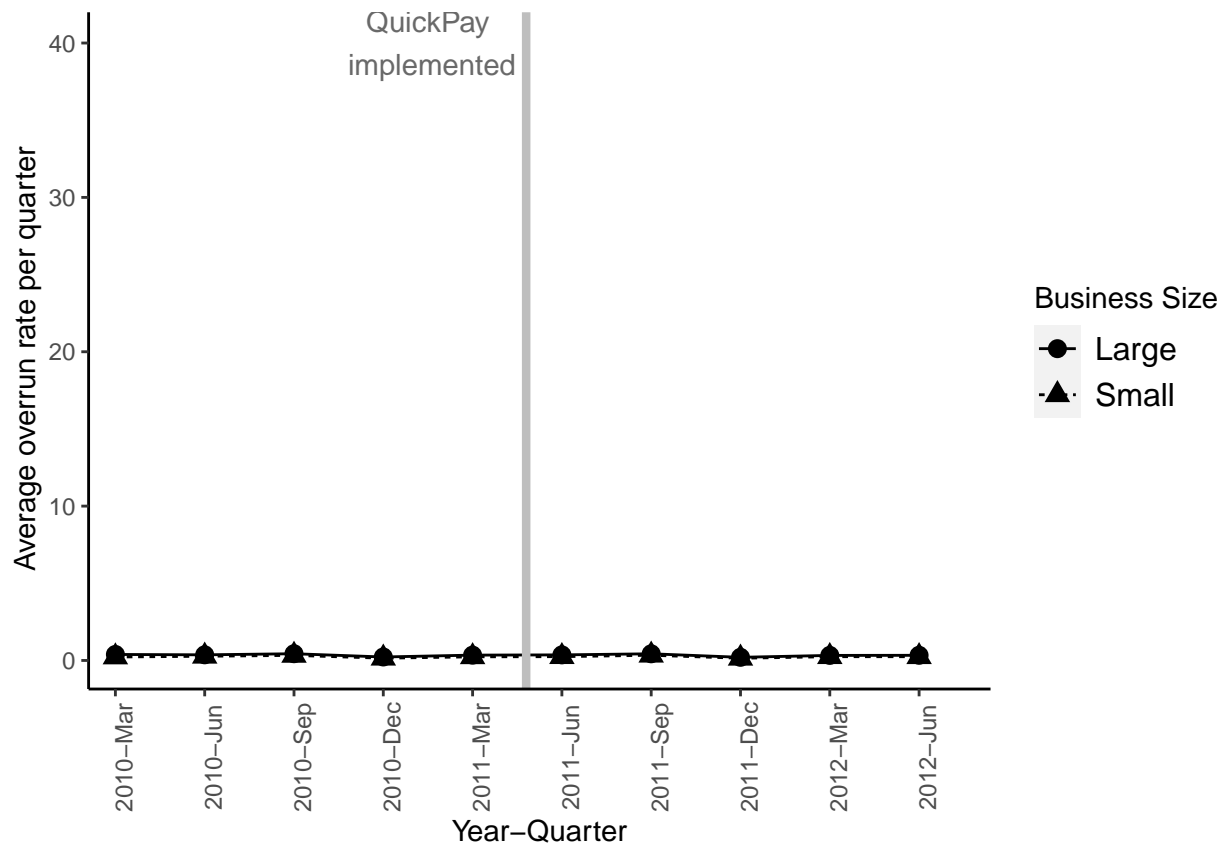
Each observation is a project-quarter.

SEs are robust and clustered at the project level.

6 Relative Overrun

6.1 Relative overruns over time

- Sample restricted to projects with modification zero when they first appeared in our sample.
- $RelativeOverrun_{it} = 100 \times RelativeOverrun_{it} / InitialBudget_i$



6.1.1.1 Normalized overrun

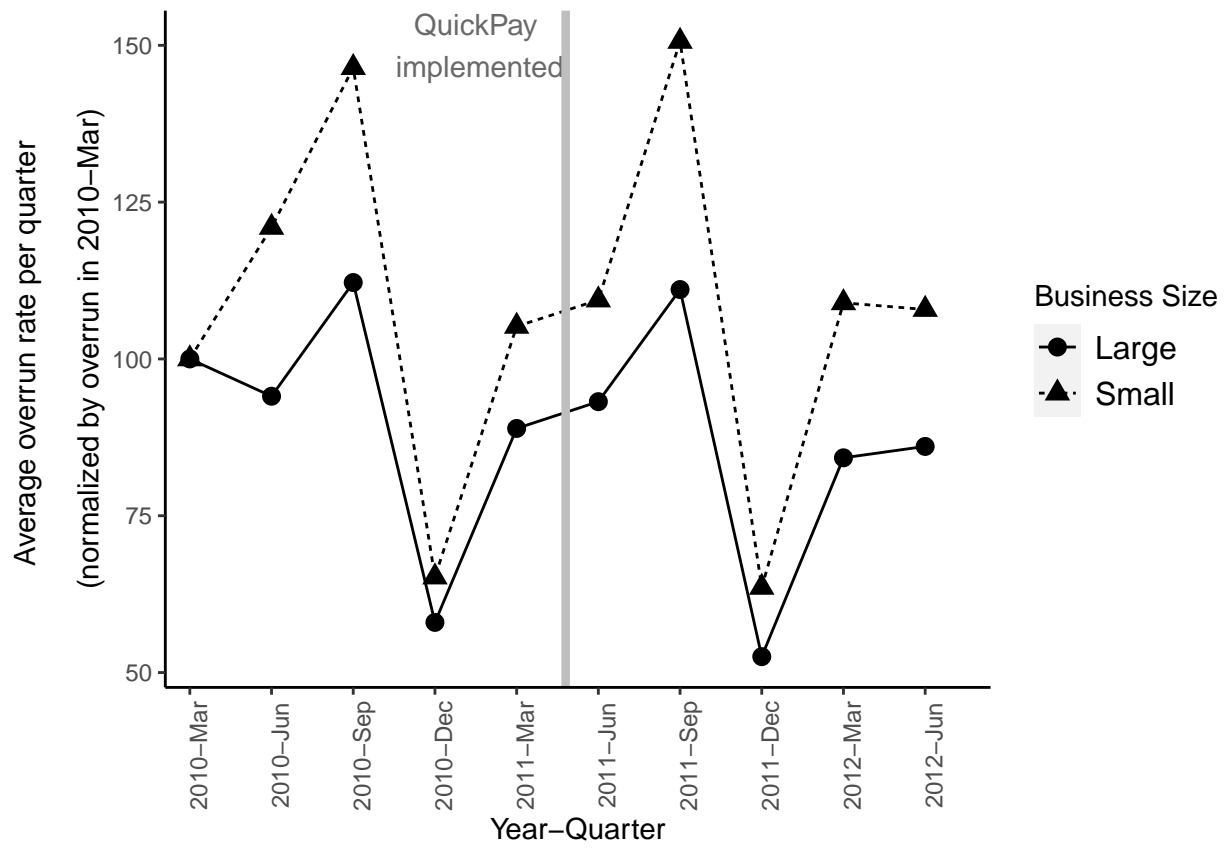


Table 3: Effect of QuickPay on project overrun rates

	<i>RelativeOverrun_{it}</i>				
	(1)	(2)	(3)	(4)	(5)
<i>Treat_i</i>	-0.10*** (0.01)	-0.09*** (0.01)	-0.09*** (0.01)	-0.01* (0.01)	-0.01 (0.01)
<i>Post_t</i>	-0.01 (0.01)	-0.10*** (0.01)			
<i>Treat_i × Post_t</i>	0.02* (0.01)	0.02* (0.01)	0.02* (0.01)	-0.005 (0.01)	-0.004 (0.01)
Constant	0.34*** (0.01)	0.54*** (0.01)			
Duration, Bids	No	Yes	Yes	Yes	Yes
<i>Post_t × (Duration, Bids)</i>	No	Yes	Yes	Yes	Yes
Project age	No	Yes	Yes	Yes	Yes
Year-Quarter fixed effects	No	No	Yes	Yes	Yes
Task fixed effects	No	No	No	Yes	Yes
Industry fixed effects	No	No	No	No	Yes
Observations	287,530	263,488	263,488	263,488	263,488
R ²	0.002	0.01	0.02	0.10	0.10
Adjusted R ²	0.002	0.01	0.02	0.09	0.09

Note:

*p<0.1; **p<0.05; ***p<0.01

Each observation is a project-quarter.

SEs are robust and clustered at the project level.