May 15, 2020 |. The Chain |. Issue 1.0

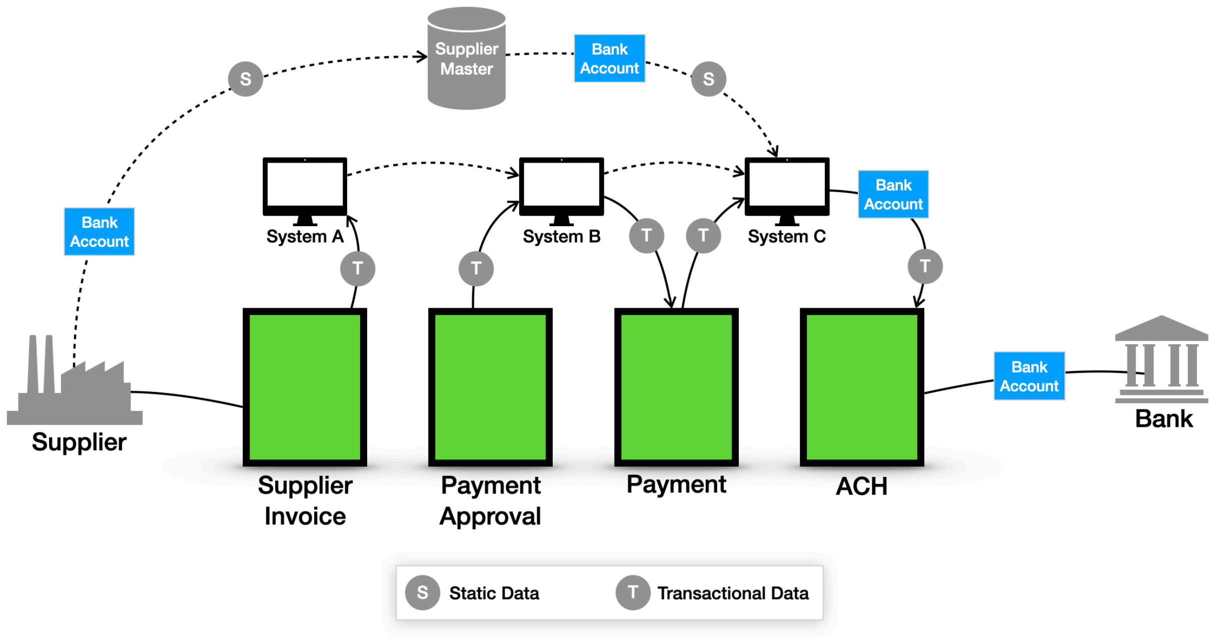
PAYMENT REDIRECT - PART I

Post COVID World

In today's COVID world, companies struggle to keep above water, or rush to pivot and take advantage of new opportunities. The new reality demands an even higher level of digitization and automation. The old days of running paper invoices, checks, deposit slips, lane lines, faxes, and lengthy reports are behind us.

New tools, including robotic process automation (RPA), internet of things (IoT), artificial intelligence (AI) and machine learning (ML), are here to stay and grow exponentially. All business communications and transactions are now handled electronically. Online customer's and supplier's orders and invoices are now digital documents, serviced by employees working from home.

Let us review a high-level payment from its creation to delivery. The diagram below demonstrates a typical case of a Supplier providing an invoice for products or services they offer. We are going to focus on what happens to the Banking information as it travels from the supplier to the company systems and finally used for payment delivery.

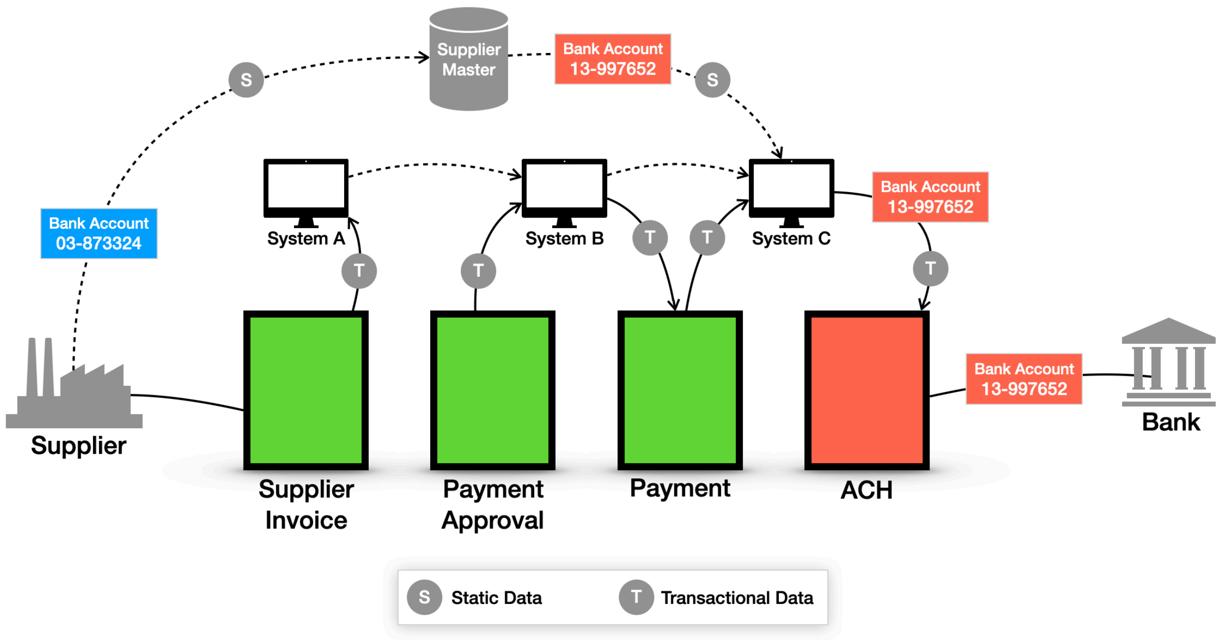


Before any payments are issued, all of the supplier information is entered into the "Supplier Master," inside a master data management system. Notice that such data changes are considered to be "STATIC DATA" manipulations. These are executed seldomly and typically do not change much over time.

On the other hand, a chain of transactions from its creation to the delivery of a payment, is considered to be "TRANSACTIONAL DATA." Later, we are going to get into this topic a bit deeper in Part II.

Use Case #1 - Temporary Supplier Master override

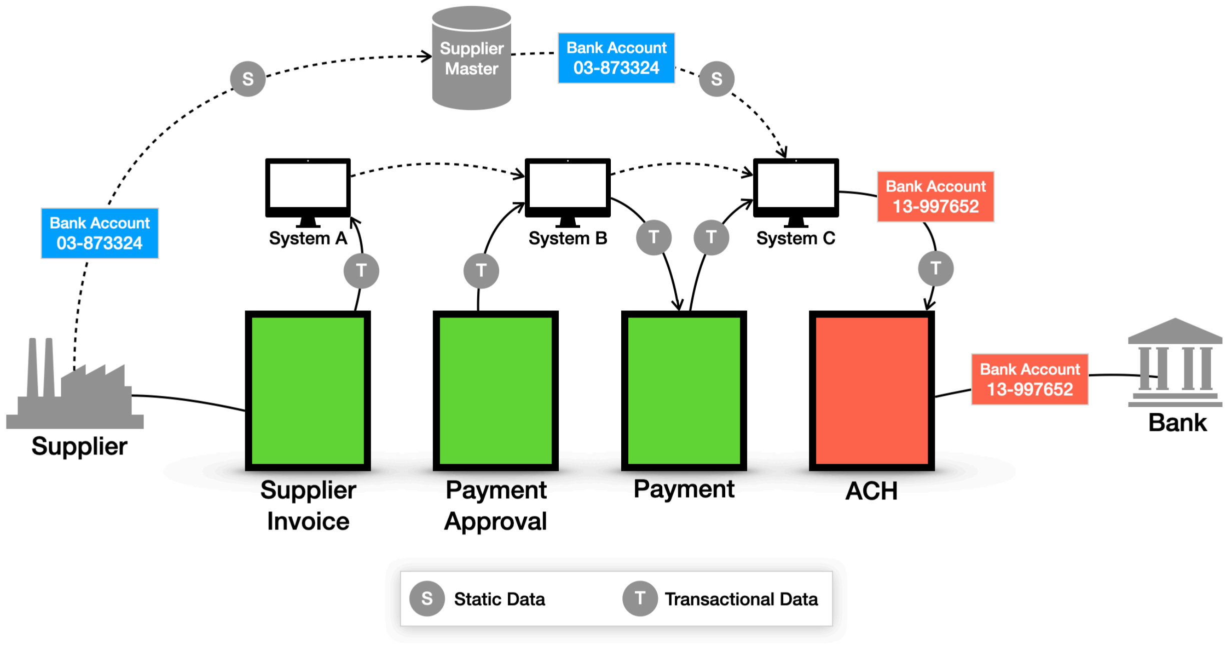
Cause: The insider here, is typically a clerk that does a lot of Master data management tasks. Their job is to update the Supplier Master. But this time, they changed the correct bank account: 03-873324, to a fraudulent one: 13-997652, just in time for the creation of the check batch. As soon as the check batch is completed, the clerk updates the Supplier Master back to the original bank account.



Effect: The incorrect account trickles down to the ACH batch system and makes it to the Bank, where the payment is issued to the fraudulent account. The payment is redirected!

Use Case #2 - ACH batch override

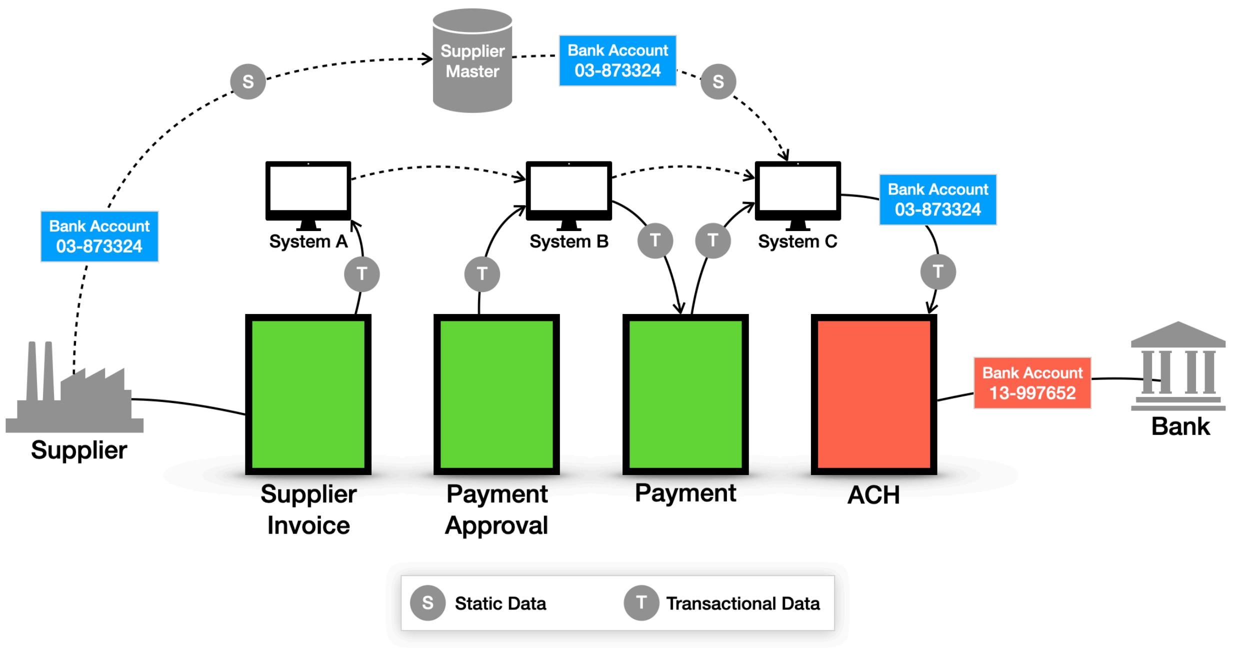
Cause: The insider here, is typically an IT system operations associate that does a lot of maintenance and system bug fixes for the ACH batch process. Their job is to make sure the process of compiling ACH records goes smoothly. After fixing the process's output several times over the last five years, it is very easy for the associate to make any changes. But this time, they change the correct bank account: 03-873324, to a fraudulent one: 13-997652.



Effect: The incorrect account trickles down to the Bank, where the payment is issued to the fraudulent account. The payment is redirected!

Use Case #3 - ACH batch override

Cause: Often a banking system consists of several "green screen" legacy systems stitched together. As a result, a high quantity of glue software is written to maintain the eco-system. System integrators use quality assurance (QA) environments to test production releases. Due to budget cuts, such environments can be used as failover during critical events. But this time, the maintenance associate simply mixes things up, and lets the QA run in place of production. The QA system changes the correct bank account: 03-873324, to a different one: 13-997652 by mistake.



Effect: The Bank issues the payment to a random account. The payment is redirected!

Conclusion

Their system's access limits an insider's control over your enterprise; however, blind trust in a good player is not sufficient. On the other hand, eco-system complexity will also cause bugs to spill out resulting in losses to the bottom line. Take control of your "Core Business Data." If a detective followed a payment from initial creation to the delivery by the Bank, all use cases above would not have succeeded.