# Quote-to-Cash Is Breaking: Why Modern SaaS Needs Unified Revenue Architecture

*By Sandeep Jain, Founder/CEO of MonetizeNow*

I was talking to a Product Manager of a SaaS company a few months back and she mentioned that subscription billing was massively broken for them and hurting them badly on revenue growth. The product manager lamented that these tools are not designed for SaaS and cause massive friction in revenue operations.

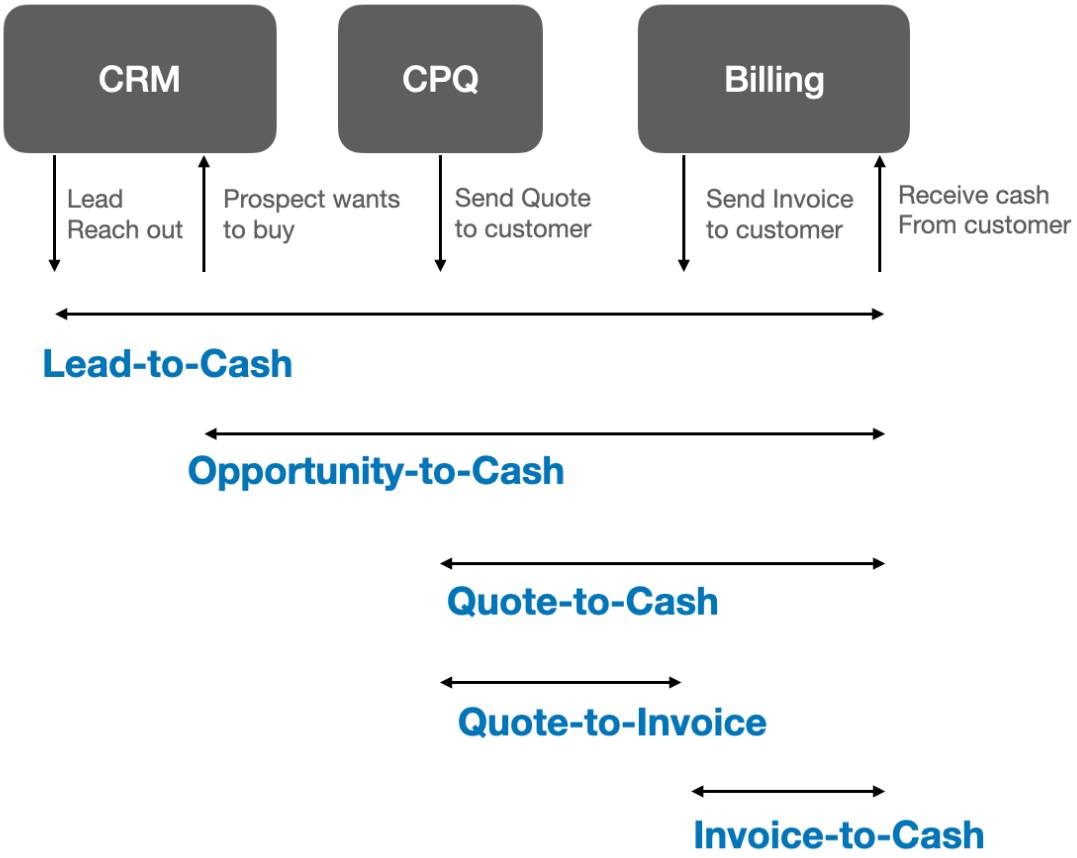
Interestingly, this is not uncommon. Irrespective of where a company is in its journey, **Quote-to-cash** is a major headache. In fact, I was told that one company spent as much as $300M (!!) in fixing Q2C and that too with less than satisfactory results.

If you are not familiar with Quote-to-cash terminology, it covers a set of steps starting from the generation of a Sales Quote and ending in receiving money from the customer and recording the transaction in the General Ledger (Accounting tool).

A related terminology is **Opportunity-to-Cash** — think of this as the superset of Quote-to-Cash terminology. Instead of the process starting from the Quoting tool, it starts a step before, i.e., in the CRM system where the intent to purchase is represented by an object called Opportunity (HubSpot CRM uses the term Deal instead).

Another related term is **Lead-to-Cash** which is the superset of Opportunity-to-Cash in that it captures the process starting yet another step before — when the customer is still a Lead (Marketing to).

Since Quote-to-cash is a mission-critical process for any company (it enables the company to earn revenue), I decided to dig deeper as to why has this become a big problem for everyone. This article presents a summary/analysis and towards the end a proposal to solve the problem based on 100+ conversations with customers/system integrators over a period of 3 months.




## The Quote-to-Cash Process: Step by Step

1. The customer intends to purchase a product captured as an Opportunity/Deal in the CRM product.
2. From there, a sales rep uses a tool called CPQ (Configure Price Quote) to configure the list of products/services the customer wants, then pricing it (this usually involves giving discounts or doing price increases for multi-year deals), and then finally generating a professional-looking (usually PDF) Quote. The sales rep can generate multiple quotes to reflect different pricing scenarios.
3. Once the customer agrees to a particular Quote, the sales rep sends the Quote to the Customer for eSign, and finally, the Opportunity in the CRM is marked as closed.
4. At this stage, the second half of the Q2C process kicks in. When the Opportunity in the CRM is closed, an email notification is sent to a few teams — usually Ops and Billing teams.
5. The job of the Ops team is to provision the system for the customer; they figure out what licenses, subscriptions, entitlements, etc., have to be turned on in the product to fulfill what is promised to the customer. This process is surprisingly predominantly still manual.

6. The Billing team uses the signed PDF information and does a swivel-chair manual data transfer to the Billing system. This involves creating customer information in the Billing system, copying the discounts/pricing line-by-line, and generating an invoice from the Billing system. It is an error-prone and challenging job.
7. Finally, the customer receives the invoice and then pays the vendor either through a credit card or ACH (or whatever payment method is prevalent in that geography).
8. Depending on how the payment is set up, either the invoice is automatically marked as paid, or the billing person has to look at the bank statement and close the invoice manually by doing a reconciliation.
9. Finally, the invoice data and status is passed to the downstream accounting system, which records the invoice amount in the correct ledger.

## Urgent Enterprise Requirements

Enterprises across all industry verticals have the following urgent requirements:

1. Transition to subscriptions and consumption-based business models
2. Ability to offer new bundles in hours, not weeks or months
3. Enable self-serve (also called Product-led-growth or PLG)

Unfortunately, the existing Quote-to-cash tools have failed to meet the above requirements resulting in a broken revenue infrastructure.

| Company | Current Tools | Problems | Next Steps |
|---|---|---|---|
| *recent IPO, Hardware* | Salesforce CPQ, Zuora | CPQ - no real concept of guided selling, large quotes (>200 lines) need to be broken up, APEX errors, no API support. Zuora - APIs and connectors suck, too slow -- around 1K test accounts in sandbox & taking 1hr+ to run, trying to install a new tax collector in sandbox...doesn't work. Too long for provisioning new SKUs. | Build own CPQ. Zuora alternative not clear. |
| *SaaS* | Oracle CPQ, Was using Zuora earlier | Zuora doesn't support bundles. Amendments challenging. Not happy with CPQ either; too complex for our requirements. | Starting building own billing system |
| *recent IPO, SaaS + some hardware* | Building own billing for last 3 years. No CPQ. | Considered Apttus Billing but it required Apttus CPQ. Other billing solutions inflexible to extend. Invoicing needs are complex (parent/child, view vs pay). Complex pricing... customer adds licenses later, with tiered pricing they are supposed to pay less but are charged the same... customers dispute, fixing requires manual labor, refund and unhappy customers. No one understands enterprise billing. No one handles hardware/services/subscriptions on one quote. | Execs started questioning internal tool. Evaluating vlocty CPQ & not Apttus/Model-N |
| *Very large Finance Market data enterprise.* | Salesforce CPQ, SAP Billing | Problematic, painful and expensive problem. 40K customers, 200 countries, 15 currencies/languages. Largest subscription $200M annually. Roughly 10-13 systems in overall Lead-2-Cash flow. Spent >$10M Deloitte. As soon as you look at the process in separate parts, you create bottlenecks requiring translations, managing discrepancies. I need one system to handle Q2C. Self-serve is a HUGE need. Needs to be easily extensible. | Looking for alternatives but none seem to be available. |
| *Very large SaaS* | Internal CPQ /Billing (some SAP) | Business growing too fast for internal tools to handle. Evaluated Apttus but didn't work. Need a scalable approach to support subscriptions (renewals, up-sells and cross-sells) | Evaluating outside options. |
| *very large hardware* | Apttus, Salesforce CPQ. Zuora | Very unhappy. Requirements -- dynamic pricing, dynamic bundling. Zuora fails to meet requirements. Expressed investment interest. | Exploring building their own. |
| *very large hardware* | Salesforce CPQ. | Hired top consulting firm only to fire them 4 months later. Want to transition to subscriptions but struggling to find the right product. | SVP level priority to fix. |
| *very large hardware + SaaS* | Salesforce CPQ/Billing | Basic amendment scenarios / bundles not handled well. Salesforce CPQ treats everything as a contract and not everything can be changed. Data in/out is an issue. Will require lot of extensions => time and $$$. | Not sure what to migrate to. |
| *Healthcare* | Apttus CPQ. Own Billing. | Gap between CPQ & Billing. Had to create own billing system that matches usage against existing contract. No product does usage-based billing. Issue in maintaining & managing internal systems. | Open to pilot promising billing product |

## Summary of Problems

- **Pricing and packaging changes take months instead of hours —** The primary reason for this delay is that the CPQ and Billing tools have different product catalogs and doing any pricing/packaging requires making painful changes in the product catalog which usually involves lengthy professional services implementation.

- **Enabling the self-serve channel is massively painful —** Existing tools were not designed for PLG so anything self-serve is a massive DIY effort.
- **Reseller channel workflow is still manual —** Resellers send a request to quote to an employee (usually a channel sales manager) who then generates the Quote and sends it across to the reseller over email. Reseller invoicing is also manual.
- **Moving customers across different sales channels is painfully manual —** e.g., from self-serve to sales-led/CRM (Expansion), or moving from sales-led/CRM to self-serve (Contraction).
- **Invoicing is surprisingly still manual and error-prone —** Billing teams do swivel-chair data transfer between the CRM and the Billing System.
- **Supporting Amendments is a painfully manual process —** Even though amendments are the basic building blocks of the SaaS land-and-expand business model. CPQ and Billing systems don't share the concept of a sales 'Contract,' meaning each amendment is treated as a new contract, creating chaos at renewal time.
- **Proration challenges —** Proration is the by-product of Amendments. When an upsell happens mid-month, determining whether to prorate based on days remaining, whole calendar months, or partial months is where Quoting and Billing systems go out of sync.
- **Renewals are massively painful —** Due to the fact that amendment data is spread across multiple disconnected deals. For low ACV deals, this should be a completely automated process instead of a manual one.
- **No good system of truth for Contracts —** Some CPQs don't even have the concept, and then Billing systems don't understand it, creating massive issues in managing SaaS business and making revenue recognition unnecessarily painful.
- **Outdated and incompatible product catalogs —** Across CRM/CPQ/Billing/Accounting systems, leading to data duplication and complicated integration work.
- **Analytics and reporting become complex —** Because data is not clean (garbage-in, garbage-out).
- **Enabling usage-based billing is a massive challenge —** Traditional billing systems can't handle real-time metering, and standalone usage systems require a separate Billing system, creating painful integration requirements.
- **Existing billing systems can't support the self-serve channel —** Most enterprise billing systems can't support the concept of free users or try-and-buy models.
- **Broken APIs —** Existing tools were designed to be UI-heavy with little to no focus on API design.
- **No good Sandbox support —** Existing CPQ and Billing tools have poor sandbox support — it takes a lot of cost and time to create one.
- **No mobile-centric experience —** Doing approvals quickly, seeing the renewal pipeline, customer lifecycle data — essentially the ability to see key business data quickly.
- **Inherent lack of scalability —** If the Quote has more than a few offerings, CPQ slows down leading to sales rep frustration. Billing systems are also known to

crack under pressure when thousands of invoices need to be generated towards end of month.
- **Implementation is massively painful —** Customers routinely spend thousands of dollars just on implementations. For every dollar spent on the tools, customers spend $3–$4 in connecting the tools.

## Why has Quote-to-Cash Become So Complex?

### Non-SaaS to SaaS Transition

The biggest change in the last few years is the transition from selling one-time products to subscriptions and now consumption-based models.

The one-time selling motion of Non-SaaS had the following key characteristics:

- **Each transaction was terminal —** no upgrades/downgrades in the future. If you need to buy more, you do another transaction.
- **Information flow was one-way —** the rep generated the quote which was then used by the billing teams to generate an invoice. There was no feedback loop from Billing to CPQ.
- **Transactions were low-volume and high-dollar —** an artifact of the sales-led sales process.

***Then came subscriptions and Usage and all the assumptions of the Non-SaaS world got turned on their head:***

- **Land-and-expand motion —** The core thesis of SaaS is that no transaction is terminal. Instead, there is a lifecycle — the customer buys some seats at the beginning of the contract period (land) and then comes back to buy more during the contract period (expand). Existing CPQs were not designed around this concept.
- **Self-serve channel —** High-volume, low-dollar transactions that rule out having sales reps do quoting. Enterprises are forced to build and manage two parallel Quote-to-cash workflows — one for direct sales and another for self-serve.
- **Complex product packaging/pricing —** Good/Better/Best plans with several add-ons require the product catalog to support the concept of time (subscriptions), bundling, and consumption-based models — essentially a rewrite for traditional CPQ and Billing vendors.
- **Rapid pricing and packaging changes —** New GTM models need to be quickly implemented in both CPQ and Billing systems. Since underlying product catalogs are disparate, any small pricing change becomes a massive effort.

SaaS also brought other related issues to the forefront:

- **CRMs designed for thousands of accounts, not millions —** Salesforce CPQ cannot handle quotes of more than 200 lines, requiring quotes to be divided into multiple quotes.

- **Guided selling is limited to manual Q&A —** Rather than context-driven (customer, country, vertical) or prior purchase history. Generative AI can be massive here.
- **Consumer UX and mobile-centricity were largely ignored.**
- **Enterprise use cases not accounted for —** Most vendors took a mid-market approach early on, resulting in inflexible systems with poor scalability, dysfunctional APIs, broken reporting.
- **Existing products were created more than 10 years back —** Most seem to be running on outdated software architecture — lack of scaling/performance, weak APIs, poor data extraction support.

> ***Fixing these issues cannot be done using a bolt-on fix-it approach. Re-architecting the core engine is essentially creating a new product. Instead, it requires a thought-through first-principles-based software design from day one.***

## What Are Existing Vendors Doing?

Either adding bolt-on products or doing acquisitions — Apttus did CPQ first, Billing later. Zuora did Billing first, CPQ later. SAP acquired Billing first then CPQ (CallidusCloud, $2.4B, Jan 2018). Salesforce acquired CPQ (Steelbrick, $360M, Dec 2015).

All of these approaches are not working — acquisition merely puts the different tools under one vendor; however, the core issue of disparate product catalogs across these tools still remains.

The multiple-product approach (whether in-house or through acquisition) introduces immense bottlenecks in data flow requiring painful data translations resulting in brittle connections and never-ending maintenance.

## How Are Enterprises Dealing With This Now?

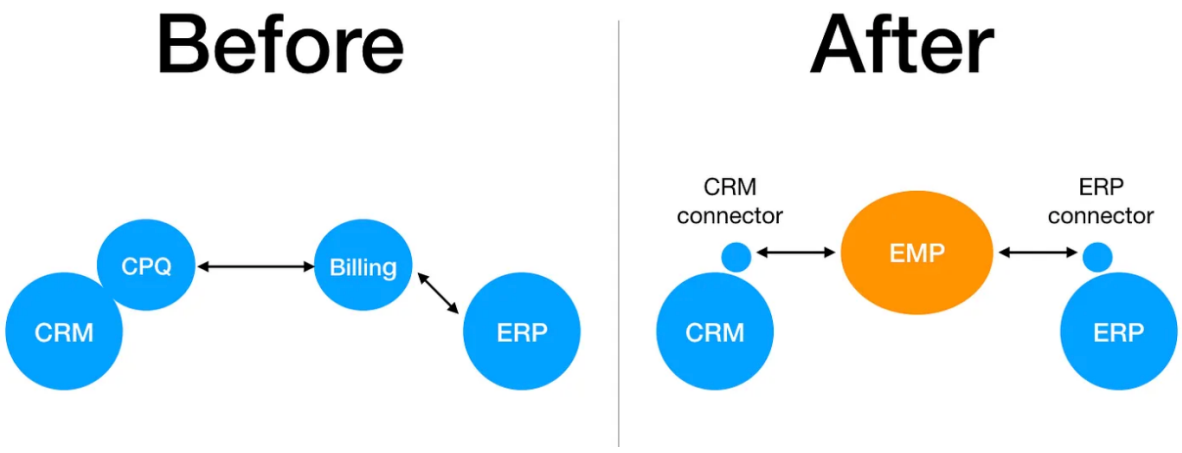
They are forced to do either of the following:

4. **Buy existing products and hire an army of System Integrators —** to somehow make the products work together by undertaking complex integrations. Roughly 10–20% of such implementations simply fail, and others that succeed end up creating brittle connections. Change management becomes a huge pain (both time and cost) and agility is completely lost. One company spent $300M (!) on a CPQ transformation project with a less than satisfactory outcome.
5. **Build own tools —** Everyone from large to medium enterprises knows this is not a long-term strategy. Creating and maintaining systems like CPQ/Billing is a complex undertaking and early benefits are quickly overtaken by long-term maintenance costs, lack of product direction, and budget cuts.

## What's the Fix?

- **First and foremost, the Pricing engine, Product catalog, and Subscriptions need to be in ONE SYSTEM.** The traditional way of putting Pricing and Catalog in CPQ and Billing in ERP separately worked well for one-time orders. With subscriptions, 'order' is not a terminal entity — it lives on with amendments. The best way to handle that is by combining subscription handling (quoting, billing, amendments, renewals), product catalog, and pricing in one system.
- **Second, think Enterprise-architecture Day 1.** That means building a platform vs an application. Make data in/out extremely easy, provide rock-solid connectors to popular CRM/ERP solutions, focus on UX, easy extensibility, mobile-centric, run on independent cloud (not force.com).
- **Third, cater to self-serve — the future of enterprise selling.** Provide an Amazon/Atlassian-like buying experience. According to Forrester, 74% of B2B buyers prefer to buy online vs through channels. PLG is the name of the game.
- **Fourth, put data science and ML to good use.** Q2C workflow sees financial, customer, and machine data — the perfect place to apply ML driving up-sells, cross-sells, and renewals. Also, tying product usage directly to customer portal to drive auto-upsell notification: 'You are ready for an upgrade.'
- **Fifth, rethink customer success:** Tying revenue to a successful implementation instead of just the initial sale. If a lot of value is created for the customer, it will be a huge win-win proposition.

## Enter… Enterprise Monetization Platform




Key benefits of a unified Enterprise Monetization Platform:

- No need to buy and manage separate Billing and CPQ products — do Subscription quoting, billing, usage, renewals, and amendments in ONE Platform.

- Usage-based Billing — send usage directly from your product to the platform. It will meter in real-time and generate an invoice that combines subscriptions and one-time products (billed in advance) and usage (billed in arrears) in one invoice.
- True self-serve — customers can do amendments and renewals automatically with standard discounts.
- CPQ interface (web + mobile) that sales will love to use — helping close deals as fast as possible.
- Provision new sales bundles in a matter of hours, not weeks.
- Ability to create dynamic bundles — no more SKU proliferation.
- Combine hardware and software in one quote.
- Subscription amendments so easy they can be self-served.
- Go live in under 1 month (post requirements gathering).

## Summary

Quote-to-cash has become increasingly complex as B2B SaaS companies adopt hybrid GTM motions, usage-based pricing, self-serve, enterprise contracts, renewals, amendments, and evolving RevRec requirements. The challenge is not a lack of tools — it is that CPQ, Billing, Metering, and RevRec often operate in silos, creating fragile handoffs, billing errors, manual reconciliations, and unreliable revenue data.

The future belongs to companies that unify revenue architecture around a single, contract-centric data model. When quoting, billing, usage, and revenue recognition operate from the same source of truth, teams can move faster, launch new pricing with confidence, improve customer experience, and unlock AI's full potential.

Unified quote-to-cash is no longer just an operational improvement; it is a strategic advantage.