



GENROCKET SOLUTION GUIDE

X12 EDI Test Data Automation

Self-Service Synthetic Data Generation for Testing EDI Healthcare Systems

Key Take-aways for QA Professionals:

- Only 20% of the 7 billion annual health insurance claims are “clean” leaving 80% with errors and omissions in patient/provider data, treatment codes or EDI formatting.
- Health insurance payers are at financial risk for overpayments, duplicate payments, fraudulent claims, and penalties for late payments.
- Using private patient information during software testing poses a HIPAA violation risk and heightens the need for test data that is 100% secure.
- COVID-19 will drive up the total volume of claims and the value of payments to providers creating a need for highly accurate and efficient claims processing.
- The full automation of paper transactions promises to unlock \$13B of industry-wide savings driving demand for increased test automation and self-service test data generation.

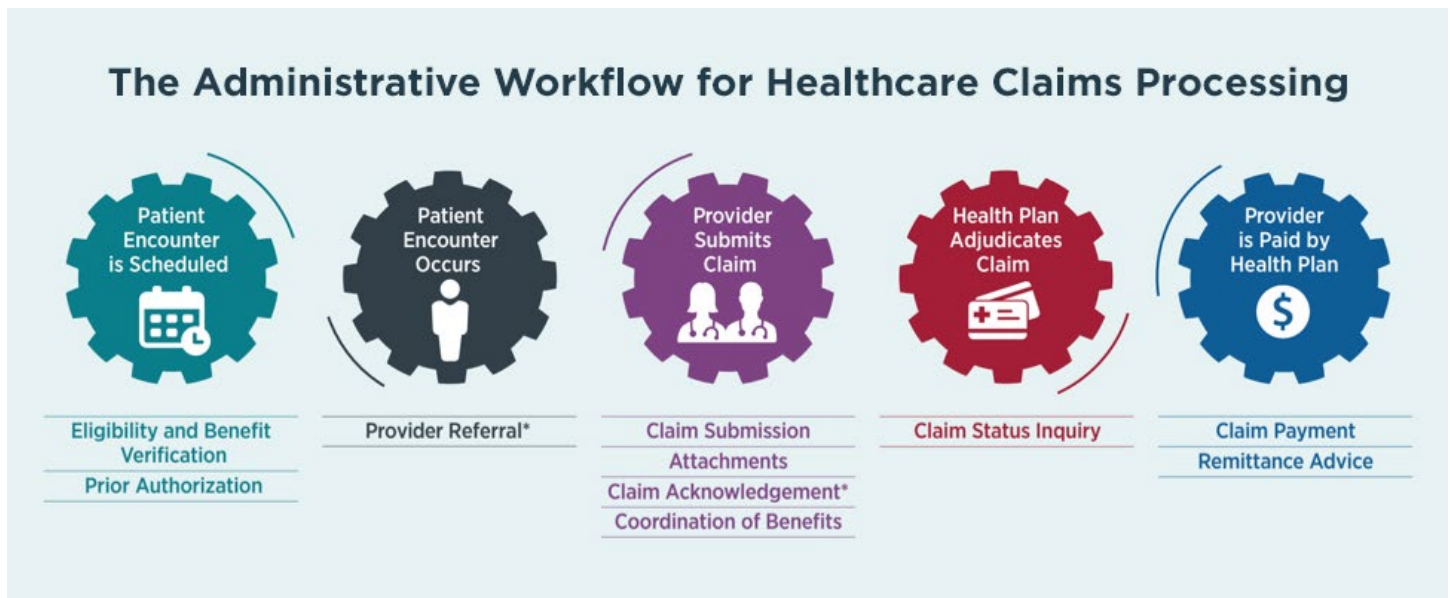
Health Insurers Face Financial and Security Risks

For public and private health insurers in the US, ensuring the quality and efficiency of their information systems is critical to controlling cost and risk. According to [CAQH](#), the cost of administering 7 billion electronic healthcare claims reached \$40 billion in 2019. While 95% of 837 claim submissions are now electronic transactions, many aspects of the administrative workflow are far less automated. CAQH estimates an additional \$13 billion in annual savings can be realized with full automation.

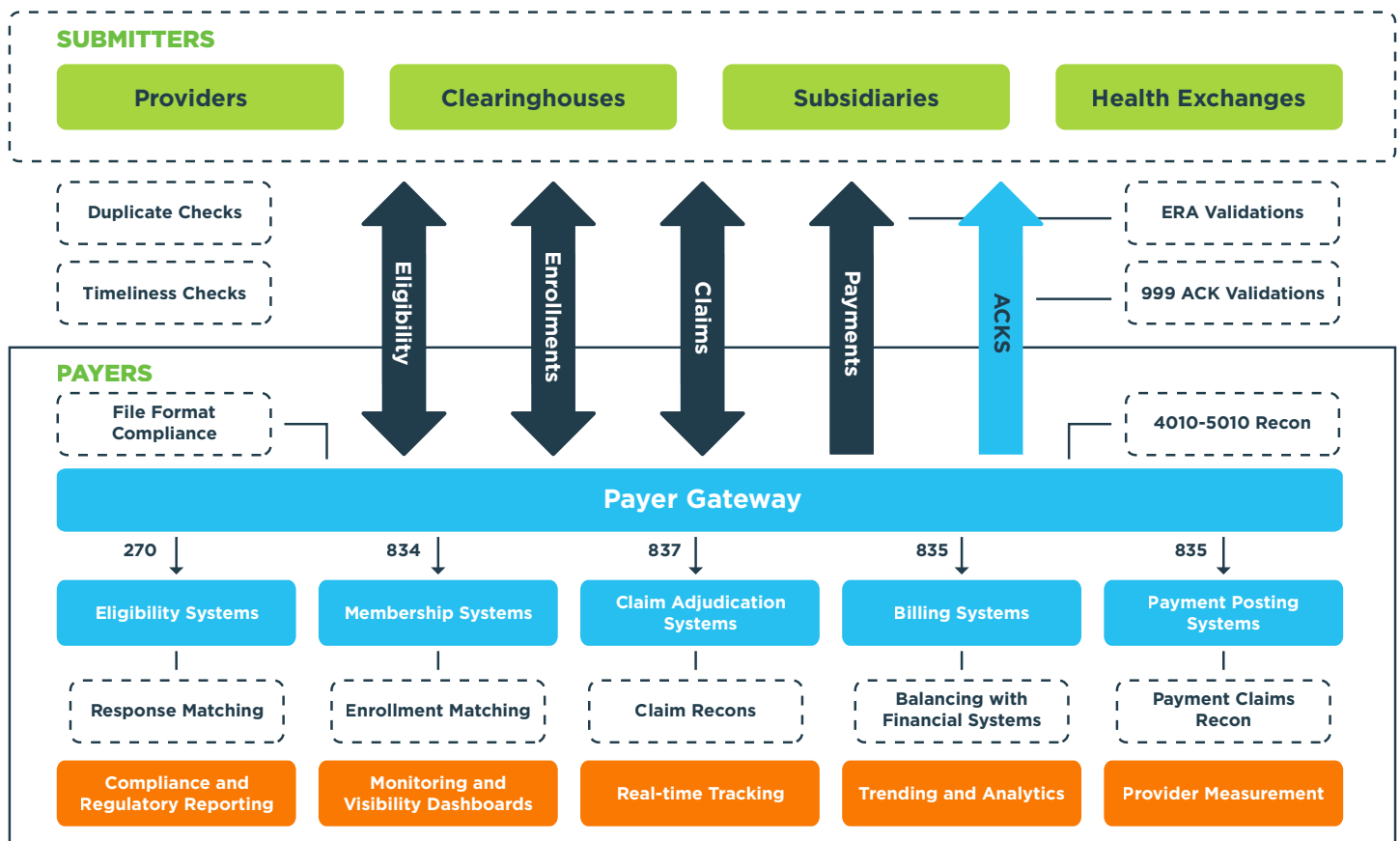
Manual procedures yet to be fully automated as electronic transactions include:

- Member enrollment
- Eligibility and benefit verification
- Prior authorization
- Provider referral
- Claims attachments
- Coordination of benefits
- Claim status inquiries
- Claims payments
- Remittance advice

The ASC X12N EDI Standard defines the structure and content of the electronic transaction sets used to automate the end-to-end administrative workflow for electronic claims processing.



This is a simplified view of the process. In reality, electronic claims processing is a complex flow of information with multiple branches and decision points connected by a network of distributed systems. Some of these systems are operated by payers, some by providers and some by third party administrators and clearinghouses. Below is a diagram of a typical healthcare claims processing ecosystem.



Connected to this ecosystem are several types of providers that comprise the patient's health care team, such as physicians, dentists, physical therapists, psychiatrists and chiropractors. The types of facilities where services are performed are equally diverse. They include hospitals, clinics, private offices, laboratories, pharmacies, emergency care, urgent care, and ambulance services. With so many moving parts, it's easy to understand the huge opportunity for errors. Studies indicate that electronic claims are riddled with errors with 4 out of 5 bills having some form of inaccuracy. Depending on the nature of the errors, this can lead to overpayments by insurers or underpayments to providers.

80% of health insurance claims are inaccurate and many errors lead to overpayments, duplicate payments or late payments by insurance companies.

The complexity of medical coding is often to blame. In healthcare claims processing, there are 3 different categories of billing code systems. **CPT** (Current Procedural Terminology) codes are maintained by the American Medical Association (AMA). **HCPCS** (Healthcare Common Procedures Coding System) codes are administered by the Centers for Medicare and Medicaid Services (CMS). And **ICD-10-CM** (International Classification of Diseases, 10th Revision, Clinical Modification) codes are maintained by the National Center for Health Statistics (NCHS) and used to classify diagnoses and disorders.

Thousands of medical codes across multiple code standards are a constant source of billing inaccuracies in the healthcare system.

The healthcare workers who assign these codes are often inadequately trained on their use. While certification programs exist, no formal education is required to be a medical coder. The use of inaccurate coding is mostly accidental, but sometimes deliberate. The [National Health Care Anti-Fraud Association](#) believes that losses from healthcare fraud could be as high as \$300 billion per year, or 10% of total healthcare spending.

Some examples of fraudulent billing practices include:

- Upcoding – Billing for more expensive services than actually provided
- Unbundling – Billing for each step of a procedure as if they were separate procedures
- Phantom billing – Charging for unnecessary procedures or services that did not take place
- Identity theft – Submitting falsified claims through the use of stolen member identities

Identity theft is the direct result of stolen medical records. The healthcare sector saw a whopping 41 million patient records breached in 2019, fueled by a 49 percent increase in hacking, according to the [Protenus Breach Barometer](#).

Finally, if payments are not made in a timely fashion, insurers may be subject to “Prompt Payment Laws” as regulated by the insurance commission for each state. These laws often come with interest and penalties charged to payers for each late payment. An efficient claims processing system will keep up with the escalating volume of claims and avoid these penalties.

How Quality Assurance Can Reduce Cost and Risk

Payers can minimize their financial and security risks by maximizing the quality of their internal claims processing systems. QA teams play a principle role in reducing cost, improving efficiency and avoiding a data breach by focusing on three test automation best practices.

1. Improving the Accuracy of Claims

A critical part of testing claims processing software is ensuring the accurate use of billing codes. A thorough test of application business logic requires the use of real medical codes across the adjudication process. This will provide the best defense against approving transactions with incorrectly assigned codes. Positive and negative test data should be conducted to fully exercise the code with both valid and invalid use cases and data values.

2. Ensuring the Privacy of Patient Data

Patient privacy is paramount for any healthcare organization. QA teams must take measures to eliminate private patient data from every aspect of the testing process. Data masking is a common approach used to reduce the risk. However, the possibility of a security breach is only truly eliminated with the use of synthetic data that has no connection to a patient’s real identity.

3. Fully Automating the Transaction Workflow

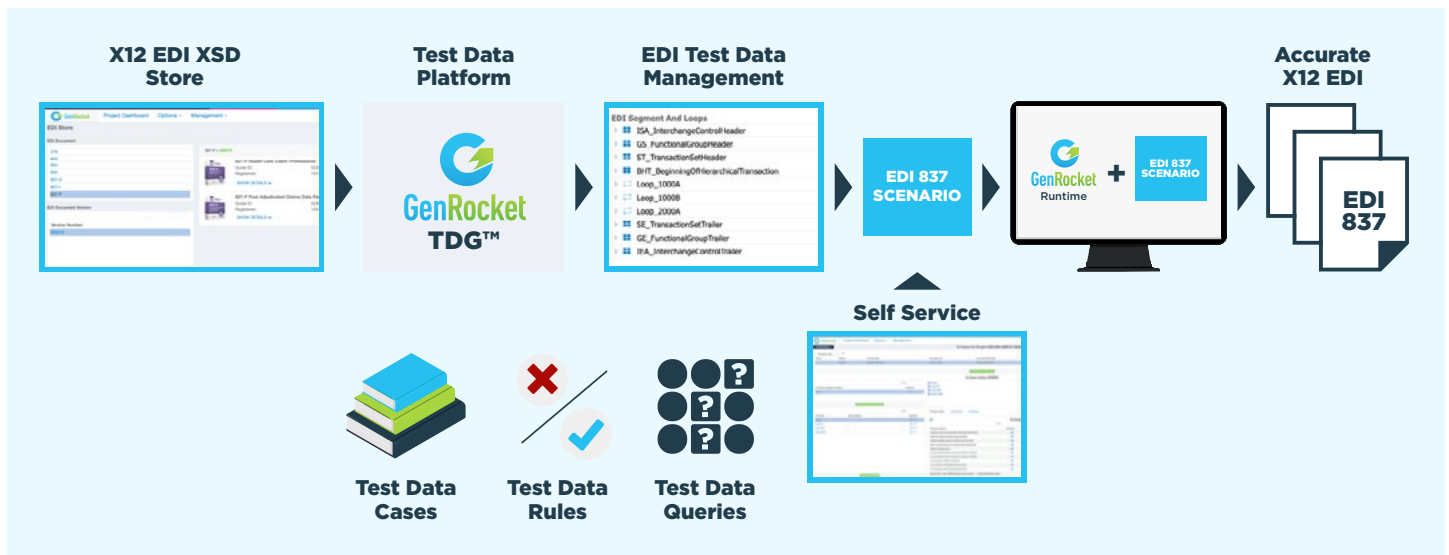
While most claims are submitted electronically, many other steps of the transaction workflow are still a manual process and rely on paper forms. The transition of claims processing to a completely electronic data interchange will drive significant cost savings while minimizing inaccuracies. Once fully automated, the QA team must validate complex workflows across the entire claims processing system.

GenRocket's EDI Test Data Automation Solution

To enable these best practices, GenRocket has developed a self-service platform for generating **real-time synthetic test data** to address the critical needs of healthcare application testing. The GenRocket **EDI Test Data Automation** solution delivers the following capabilities:

- Validate claims processing systems with fully compliant X12 EDI test data
- Customize pre-built EDI transaction sets for any payer implementation
- Blend clinically accurate production data (medical codes) with controlled synthetic data
- Generate rules-based synthetic data for testing the accuracy of business logic
- Create dynamic data for testing state transitions during complex workflows
- Self-provision test data on-demand to enable continuous, accelerated testing
- Use an intelligent API to integrate test data generation with test automation tools

The use of real-time synthetic test data in conjunction with queried production data can maximize test coverage at each stage of software integration – unit, integration, and system testing. GenRocket has developed a comprehensive EDI Test Data Automation solution that makes the process easy and automated.



The Value of Synthetic Data

There are many reasons for using synthetic data in place of, or in combination with, production data. Here are 10 reasons for implementing real-time synthetic test data generation:

1. **VERSATILITY:** Testers can design the precise data they need for each test case and eliminate, or augment, the use of production data for more control over data variety.
2. **SIMPLICITY:** Synthetic data is generated on-demand, eliminating the need to manually create data with the data variety needed for a given test case.
3. **SCALABILITY:** Peak loads for any type of traffic are easy to simulate with millions or even billions of realistic synthetic transactions.
4. **SECURITY:** Production data can be masked to protect private data, but synthetic data replaces it without accessing or potentially exposing sensitive patient information.
5. **SPEED:** Copying, masking and subsetting production data is a centralized and time-consuming process as opposed to generating synthetic data with a self-service platform.
6. **EFFICIENCY:** Synthetic data generation produces a fresh copy of data for each test run, eliminating the need for data refresh and test data management.
7. **COST:** Synthetic data is generated locally and dynamically during each test and purged when the test is complete requiring far less infrastructure for networking and storage.
8. **COVERAGE:** Synthetic data can be designed for positive and negative testing to fully exercise code and expose potential defects.
9. **ACCURACY:** Synthetic data can be selectively blended with real production data to ensure clinical accuracy when processing medical codes for adjudicating insurance claims.
10. **VALUE:** Integration with test automation tools and frameworks allows for more testing with fewer manual steps and increases the return on your test automation investment.

A Flexible Self-Service Platform

The GenRocket **EDI Test Data Automation** solution is designed to generate controlled, accurate, synthetic test data that fully conforms to ASC X12N EDI specifications. It allows testers to design the precise data needed for testing in any volume. The platform includes a self-service portal, called **G-Self-Service** that provides access to synthetic EDI test data at any stage of the Agile SDLC. And GenRocket has developed several example implementations for the most common EDI transaction sets to accelerate the time to deploy test automation across the claims processing workflow.

Here is an example of a X12 EDI transaction format.

```
1  ISA*00*          *00*          *ZZ*84980          *ZZ*43201          *201910*2046**^*00501*100000000*0*T*:~
2  GS*BE*10-0000000*DEMO*20191031*2046*200000000*X*005010X220A1~
3  ST*834*300000000*005010X220A1~
4  BGN*00*(404) 569-6111*20191101*2046*01*1*1 *2*1 ~
5  REF*38*Q*Lorem ipsum dolor sit amet*0~
6  DTP*007*D8*20191031~
7  QTY*DT*001-01-0001*Augusta*Springfield~
8  N1*P5*The Mosaic Company*24*10-0000000~
9  N1*IN*Baker Hughes Incorporated*94*10-0000000*U *H~
10 N1*B0*Hewlett-Packard*94*10-0000000*J *L~
11 ACT*1*****1~
12 INS*N*01*001*01*A*A:0:1:1^1*AC*F*N*D8*20190902*R*1*1 *1*1~
13 REF*0F*690481189~
14 REF*1L*304281768~
15 REF*17*1~
16 DTP*050*D8*20191101~
17 NM1*74*1*Tatum*Tabetha*F*Dr.*Sr.*34*146180876~
18 PER*IP**AP*(452) 248-4646*AP*(736) 717-9726*AP*(442) 183-3980~
19 N3*2000 N Washington Dr*PO Box 1000 ~
20 N4*Jefferson City*MO*65101*KG*60~
21 DMG*D8*20001031*F*B*7:RET:0^1****REC~
22 EC*01*01*01~
23 ICM*1*20.00*28*TW*8~
24 AMT*B9*8841.80~
25 HLH*N*48*50~
26 LUI*LD*8*Lorem ipsum dolor*5~
27 NM1*70*1*Dawkins*Laci*Messinger*Mr.*Jr.*34*987644140~
28 DMG*D8*20001031*F*B*7:RET:0^1****REC~
29 NM1*31*1~
30 N3*1000 N Washington Dr*PO Box 1000 ~
31 N4*Trenton*NJ*08601*MP~
32 NM1*36*1*UZlimited.web*Georgetta*Turman*Dr.*Jr.*24*864215501~
```

GenRocket developed its EDI solution in close cooperation with the [ASC X12N subcommittee](#) to ensure the accuracy and compliance of its synthetic transaction sets. The solution makes use of XML Schema Definition (XSD) files sourced directly from the Washington Publishing Company (WPC) to model and generate EDI test data that is fully X12 compatible. The solution includes **G-Self-Service**, powerful self-service modules that allow controlled, accurate and secure X12 test data to be provisioned for DevOps and Agile environments on-demand.

GenRocket's EDI X12 solution generates test data for any category of testing:

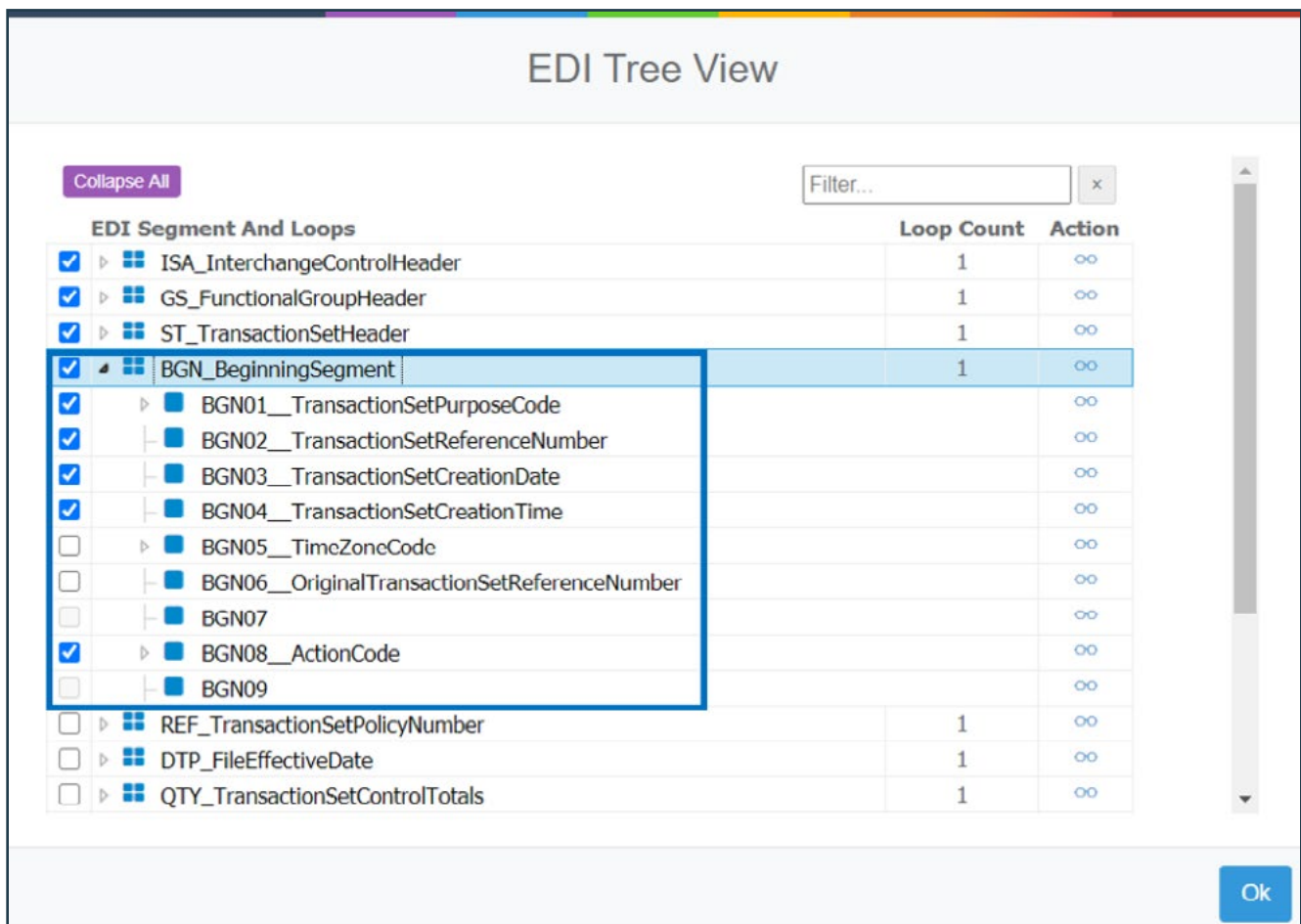
- Positive and negative testing
- Range and boundary testing
- Data permutations and combinations
- Workflow testing across multiple API's
- Synthetic data to ensure patient privacy
- Production data to inject real-world data values
- High volume data for load and performance testing

Using the **GenRocket EDI Management Dashboard**, testers have easy access to an integrated EDI Store containing the full library of EDI X12 electronic documents. With point and click access to each transaction set, testers are able to quickly create a project for generating synthetic test data based on the appropriate X12 format.

When an EDI Project is created in GenRocket, the entire transaction set is enabled. However, when testing a transaction set, not every segment, loop, and element is needed to complete each test, or to achieve a specific test case objective.

GenRocket makes it easy to create the precise test data needed for each EDI transaction type.

GenRocket users can easily customize any transaction set with a fully expandable **EDI Tree View** of the entire EDI document to edit its segments and loops. This allows fine-grained control over the structure and content of the data. This powerful editing capability makes it easy to adapt the EDI X12 standard to any specific implementation used by payers, providers and partners.



The screenshot displays the "EDI Tree View" interface. At the top, there is a "Collapse All" button and a "Filter..." search box. Below this is a table with the following columns: "EDI Segment And Loops", "Loop Count", and "Action". The table lists various EDI segments and loops, each with a checkbox for selection and a loop count. The "BGN_BeginningSegment" row is highlighted with a blue border, and its sub-items are also visible. The "Action" column contains infinity symbols (∞) for most rows, indicating that the loop count can be set to infinity.

| EDI Segment And Loops | Loop Count | Action |
|--|------------|--------|
| <input checked="" type="checkbox"/> ▶ <input checked="" type="checkbox"/> ISA_InterchangeControlHeader | 1 | ∞ |
| <input checked="" type="checkbox"/> ▶ <input checked="" type="checkbox"/> GS_FunctionalGroupHeader | 1 | ∞ |
| <input checked="" type="checkbox"/> ▶ <input checked="" type="checkbox"/> ST_TransactionSetHeader | 1 | ∞ |
| <input checked="" type="checkbox"/> ▶ <input checked="" type="checkbox"/> BGN_BeginningSegment | 1 | ∞ |
| <input checked="" type="checkbox"/> BGN01__TransactionSetPurposeCode | | ∞ |
| <input checked="" type="checkbox"/> BGN02__TransactionSetReferenceNumber | | ∞ |
| <input checked="" type="checkbox"/> BGN03__TransactionSetCreationDate | | ∞ |
| <input checked="" type="checkbox"/> BGN04__TransactionSetCreationTime | | ∞ |
| <input type="checkbox"/> ▶ <input type="checkbox"/> BGN05__TimeZoneCode | | ∞ |
| <input type="checkbox"/> BGN06__OriginalTransactionSetReferenceNumber | | ∞ |
| <input type="checkbox"/> BGN07 | | ∞ |
| <input checked="" type="checkbox"/> ▶ <input checked="" type="checkbox"/> BGN08__ActionCode | | ∞ |
| <input type="checkbox"/> BGN09 | | ∞ |
| <input type="checkbox"/> ▶ <input type="checkbox"/> REF_TransactionSetPolicyNumber | 1 | ∞ |
| <input type="checkbox"/> ▶ <input type="checkbox"/> DTP_FileEffectiveDate | 1 | ∞ |
| <input type="checkbox"/> ▶ <input type="checkbox"/> QTY_TransactionSetControlTotals | 1 | ∞ |

An "Ok" button is located at the bottom right of the interface.

A second and even more powerful way to rapidly provision EDI test data is with *Transaction Examples*, pre-built and pre-configured documents for the most common transaction sets. Whether it's an 834 enrollment, an 837 claim submission, or an 835 payment, *Transaction Examples* accelerate the provisioning process with configuration files that have the correct number and arrangement of segments, loops and elements for that particular transaction set.

G-Self-Serve | G-Cases for Project EDI-837-P-005010-X222 - 2.0

Template View | EDI | Filter

| S.No | Name | Created By | Created On | Last Modified By | Modified On | Actions |
|------|------------------|------------|--------------|------------------|--------------|---------|
| 1 | TestBusinessCase | | Aug 04, 2020 | | Sep 23, 2020 | |

Create New G-Case Suite

G-Case Suite: TestBusinessCase

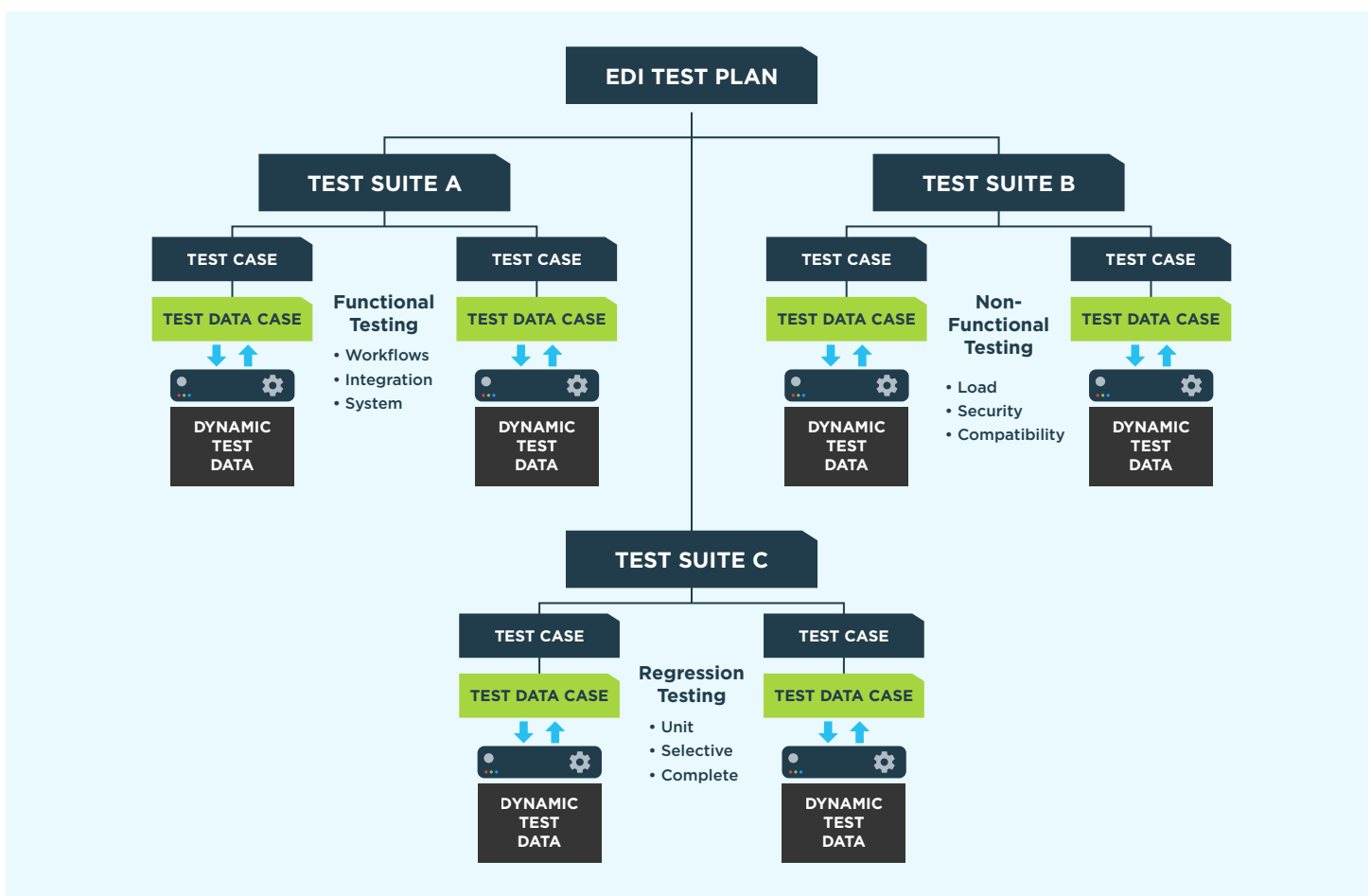
- businessScenario01 - Commercial Health Insurance
- businessScenario05 - Ambulance Claim
- businessScenario06 - Chiropractic Claim
- businessScenario09 - Anesthesia Claim

G-Case Editor | G-Rule Set | G-Queries

G-Case Category: Integration, G-Case: businessScenario01

| Domain Name | Actions | Domain Name | Loop Count | Actions |
|-----------------------------------|---------|-------------------------------------|------------|---------|
| AdjustedRepricedClaimNumber2300 | ∞ | InterchangeControlHeader | 1 | |
| AdjustedRepricedReferenceNumbe... | ∞ | V1HealthCodeInformation2300V101 | 1 | |
| AmbulanceCertification2300 | ∞ | BeginningOfHierarchicalTransaction | 1 | |
| AmbulanceCertification2400 | ∞ | V1HealthCodeInformation2300V102 | 1 | |
| AmbulanceDropoffLocation2310F1 | ∞ | SubmitterName1000A1 | 1 | |
| AmbulanceDropoffLocation2310FV11 | ∞ | V1HealthCodeInformation2300V103 | 1 | |
| AmbulanceDropoffLocation2420H1 | ∞ | SubmitterEDIContactInformation1000A | 1 | |
| AmbulanceDropoffLocation2420HV11 | ∞ | V1HealthCodeInformation2300V104 | 1 | |

The configuration files used by the GenRocket platform for generating synthetic EDI X12 test data are called *Test Data Scenarios*. Scenarios can be combined into a *Scenario Chain* to represent a complex electronic document. Scenario Chains can be combined with a *Test Data Case* to become a ready-to-execute test data design for any test case in your EDI test plan.



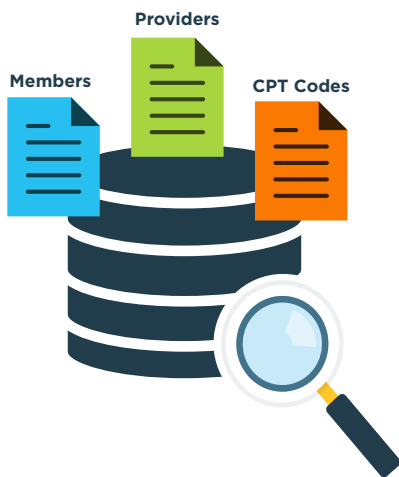
There are many ways to call a *Test Data Case* during test execution including via Jenkins, a batch file, shell script, or a scripting language. This allows test data to be generated dynamically during the testing process. Because GenRocket generates test data in milliseconds, thousands of rows of data can be injected into an application-under-test in less than 1 second. More importantly, the nature of the data can be precisely controlled to produce the combinations, patterns and permutations needed to maximize coverage. This allows any tester to provision controlled EDI transaction data for any category of functional, non-functional and regression testing.

To learn more about Test Data Cases, view our [Knowledgebase article](#).

Flexible and Powerful Self-Service Modules

As the foundation for GenRocket's self-service solution, *Test Data Cases* provides a toolset for meeting any testing challenge. Two essential components of this toolset that add power and flexibility to GenRocket's EDI solution are *Test Data Queries* and *Test Data Rules*.

Test Data Queries



With *Test Data Queries*, GenRocket can dynamically retrieve production data during test execution and blend it with synthetic data. Integrating certain enumerated production data values into the testing process can increase the clinical accuracy of the test by using real member information, provider information or medical codes for testing the claims processing workflow.

GenRocket makes it easy to blend queried production data with generated synthetic data. This provides the best of both worlds: the accuracy of real data with control over synthetic data values.

To learn more about **Test Data Queries**, view our [Knowledgebase article](#).

Blending synthetic and production data is a powerful concept. It allows for designing test data with production data that has been queried from multiple data sources to be combined with controlled and 100% secure synthetic data.

The diagram illustrates the retrieval of specific production data values, like a member ID and CPT code, to be blended with synthetic data values for a patient's name, address and birth date.

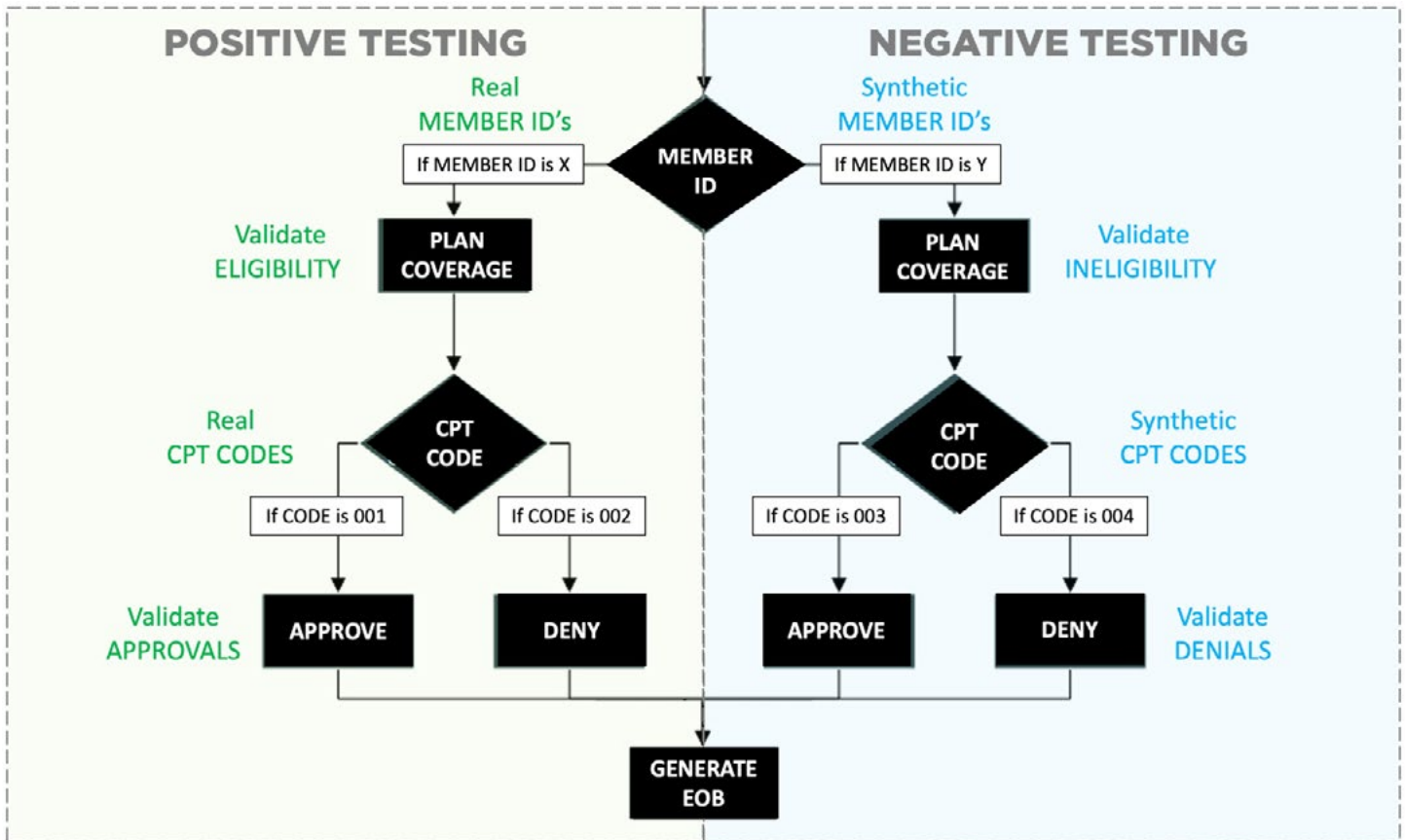
Blending Synthetic Data with Production Data



Blended test data enables your test case to validate assertions with data values that are clinically accurate while controlling the variety of data patterns and permutations.

With *Test Data Rules*, business logic can be tested using both positive and negative paths in a complex workflow. *Test Data Rules* allows the tester to configure test data based on conditional if-then-else statements in a Test Data Case used to control the test data generation process.

Accurate and Controlled Data for Workflow Testing



By combining Test Data Queries with Test Data Rules, a test case can use clinically accurate data values queried from a production data source to validate a positive test condition while using synthetic data to detect errors with a negative test condition. This allows maximum test coverage for all expected and unexpected inputs and outcomes for a given test case.



Test Data Rules are managed by the **G-Rule** self service module. *Test Data Rules* can be a simple rule set or a complex, nested arrangement of conditional logic to control all aspects of the data generation process.

The **G-Rule Set Report** provides an at-a-glance view of the rules defined for a given *Test Data Case*. Rules can be combined into *Test Data Rule Suites* to enable testing workflows with increasing complexity as code is integrated into an end-to-end system.

To learn more about **Test Data Rules**, view our [Knowledgebase article](#).

Test Data Cases and Transaction Set Examples

The self-service modules in GenRocket's **EDI Test Data Management** solution enable the rapid deployment of test automation and test data automation for EDI applications. GenRocket has created a number of pre-configured *Test Data Case Transaction Examples* within **G-Self-Service** to accelerate the test data provisioning process. These examples are 95% ready to run and require only a small degree of customization to fully conform to the specific implementation of your company's X12 EDI implementation.

The following paragraphs provide a brief overview of the most common X12 EDI transaction sets and some typical business case scenarios. Each section contains a link to GenRocket's knowledgebase where you will find a detailed explanation for each transaction set, informational videos, and instructions for how to use pre-configured transaction set examples.

X12 EDI 834 Transaction Set Examples

The 834 transaction set is used for benefit enrollment and maintenance. It provides an electronic exchange of information relating to member benefits for a given healthcare plan. Business case scenarios include:

- Enrolling an Employee in Multiple Health Care Insurance Products
- Adding a Dependent (Full-Time Student) to an Existing Enrollment
- Enrolling an Employee in a Managed Care Product
- Adding Subscriber Coverage
- Changing Subscriber Coverage
- Cancelling a Dependent
- Terminating Eligibility for a Subscriber
- Reinstating an Employee
- Reinstating the Employee at the Coverage (HD) Level
- Reinstating Member Eligibility (INS)

GenRocket provides transaction set examples for several of the above business case scenarios within the **G-Cases** dashboard which you can copy and customize to meet your specific needs.

For an overview of **EDI 834 Test Data Cases**, view our [Knowledgebase article](#).

X12 EDI 837I Transaction Set Examples

The 837I (institutional) transaction set is the standard format used by institutional providers to transmit health care claims electronically. Business case scenarios include:

- Institutional Health Insurance Claims
- Two Claims for the Same Provider
- PPO Repriced Claim
- Out of Network Repriced Claim
- Automobile Accident

Transaction set examples for the above scenarios are also available within the **G-Cases** dashboard, which you can customize to match your implementation.

For an overview of **EDI 837I Test Data Cases**, view our [Knowledgebase article](#).

X12 EDI 837P Transaction Set Examples

The 837P (professional) transaction set is the standard format used by health care professionals and suppliers to transmit health care claims electronically. Business case scenarios include:

- Commercial Health Insurance Claims
- Encounter Claims
- Coordination of Benefits Claims
- Medicare Secondary Payer Example (COB)
- Ambulance Claims
- Chiropractic Claims
- Oxygen Claims
- Wheelchair Claims
- Anesthesia Claims
- PPO Repriced Claims
- Out of Network Repriced Claims

Transaction set examples are available for provisioning 837P test data within the **G-Cases** dashboard ready for customization.

For an overview of **EDI 837P Test Data Cases**, view our [Knowledgebase article](#).

X12 EDI 835 Transaction Set Examples

The 835 transaction set is used for health care claim payment and remittance advice. It's used primarily by healthcare insurance plans to make payments to healthcare providers, to provide Explanations of Benefits (EOBs), or both. Business case scenarios include:

- Governmental institutional environment. One or more Depository Financial Institutions is involved in transferring information from the sender to the receiver.
- Managed care environment. Funds are sent by EFT to the provider's account, and the remittance data is transmitted directly to the provider.
- Report secondary or tertiary payments back to the provider.

A transaction example for provisioning 835 test data is available via the G-Cases dashboard and like the other examples, can be tailored for your organization's unique requirements.

For an overview of **EDI 835 Test Data Cases**, view our [Knowledgebase article](#).

Meeting Your EDI Test Data Automation Challenge

GenRocket's solution for **EDI Test Data Automation** can solve any test data challenge. It allows testers to provision fully compliant EDI data on-demand using a self-service portal for locally generated test data. It enables clinically accurate production data to be blended with secure and controlled synthetic data to maximize test coverage. And it provides real-time data generation, in high volume, with full customization to conform to any EDI implementation.

The EDI Test Data Challenge

Provisioning Clinically Accurate Data

Eliminating Personally Identifiable Information

Linking and Blending Real Data with Synthetically Generated Data

Adapting to X12 EDI Standards

Managing and Versioning EDI Documents

Obtaining High Volume Data for Load & Performance Testing

Most importantly, GenRocket's EDI solution enables the quality assurance best practices that reduce the cost and risk factors when testing EDI insurance claims processing applications:

1. Improving the Accuracy of Claims
2. Ensuring the Privacy of Patient Data
3. Fully Automating the Transaction Workflow

The **GenRocket EDI Test Data Automation** solution will allow your QA organization to ensure higher quality software, with greater efficiency, as you maximize your return on investment in test automation tools and technologies.