



# GENROCKET CASE STUDY

## Test Data Generation for Healthcare Markets

### BACKGROUND

Healthcare Information Technology is a large global market that is projected to grow at a rate of 13.7% annually between 2018 and 2023 and is expected to reach USD 223.6 billion in global revenues in 2023 ([Meticulous Research](#)).

This market includes a wide variety of software solutions for addressing both clinical and non-clinical applications such as:

- Patient registration, tracking and scheduling applications
- Radiology, surgery, laboratory and pharmacy management
- Medical records, billing, purchasing and payroll systems
- Any application related to in-patient or outpatient services

There is a critical need for secure and efficient data sharing between these application environments to eliminate unnecessary paperwork and to avoid the expense of custom integration systems. As a result, global standards bodies have developed an international data interchange format that allows all healthcare applications to use a common data format.

That standardized data format is called Health Level-7 or simply HL7.

### GENROCKET'S HL7 SOLUTION

A major healthcare company hired a new CIO and QA Director and as part of a company-wide move to digital it was quickly determined that they needed to improve their test data management. Their current test data provisioning process was taking days and, in order to keep pace with development, needed to be reduced to hours or minutes. With limited budget, the team set out to find a way to provision test data in HL7 format quickly, securely and cost effectively.

# THE TECHNICAL CHALLENGE

The organization had a mandate to eliminate the use of Personally Identifiable Information (PII) in their testing environment. That led them away from the use of production data and toward GenRocket's *Test Data Generation* platform that produces 100% secure synthetic test data in real-time. They needed to be able to condition and control the test data in HL7 format to support variety of test cases and generate HL7 files on-demand with any level of complexity.

## UNDERSTANDING HL7 SEPARATOR RULES

HL7 is a complex data format that defines the structure and the encoding standards for all message types, segment separators, and field delimiters allowing healthcare applications to efficiently communicate using a common data format.

The following are the HL7 recommended values:

(xOD)	Segment Separator
	Field separator, aka pipe
^	Component separator, aka hat
&	Sub-component separator
~	Field repeat separator
\	Escape character

- The segment separator is always a carriage return (ASCII 13 or HEX 0D)
- HL7 lets you choose your special characters providing they are defined in the MSH segment
- The MSH is the first segment of all HL7 messages (except HL7 batch messages)
- The field separator is the 4th character and represents the first field of the MSH segment
- The first field of the MSH is typically only a pipe (|) so counting MSH fields can be tricky
- Field 2 of the MSH (MSH-2) contains the other separator characters in the order: sub-composite, repeater, escape, sub-sub-component

Here an example of the beginning of an HL7 message:

```
MSH|^~\&|...
```

# UNDERSTANDING HL7 COMPOSITES

Each segment of an HL7 message consists of one or more composites. By default, the | (pipe) character is used to separate one composite from another. A composite can be a primitive data type (e.g. character string or a number), or can contain other composites. If a composite contains other composites, these sub-composites are normally separated by ^ characters. If a sub-composite also contains composites, these sub-sub-composites are normally separated by & characters. Sub-sub-composites must be primitive data types.

In essence, composites may descend three levels deep:

- composite
- sub-composites
- sub-sub-composites

# UNDERSTANDING HL7 MISSING SEGMENTS

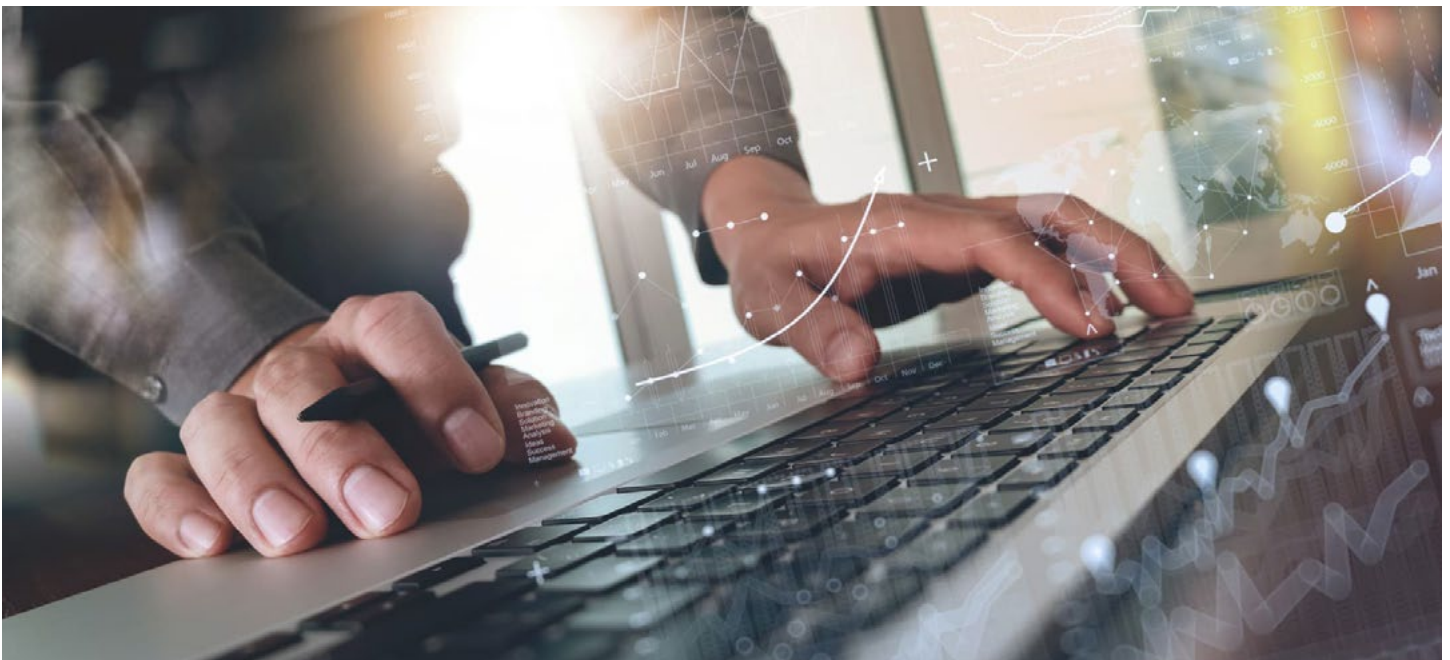
A separator identifies missing composites and its composite level determines the separator.

Here is a sample of HL7 data with missing segments:

```
PID || 0493575^^^ID 1 | 454721 || DOE^JOHN^^^^ | DOE^JOHN^^^^ | 19480203 | M || B | 254 MYSTREET  
AVE^^MYTOWN^OH^44123^USA || (216)1234567 || M | NON | 400003403~1129086 |
```

In this segment defined above, the fifth composite is the patient name, which is DOE^JOHN^^^^. The four ^^^^ characters at the end of this composite indicates that it has a total of six sub-composites, and that only the first two of the sub-composites are defined.

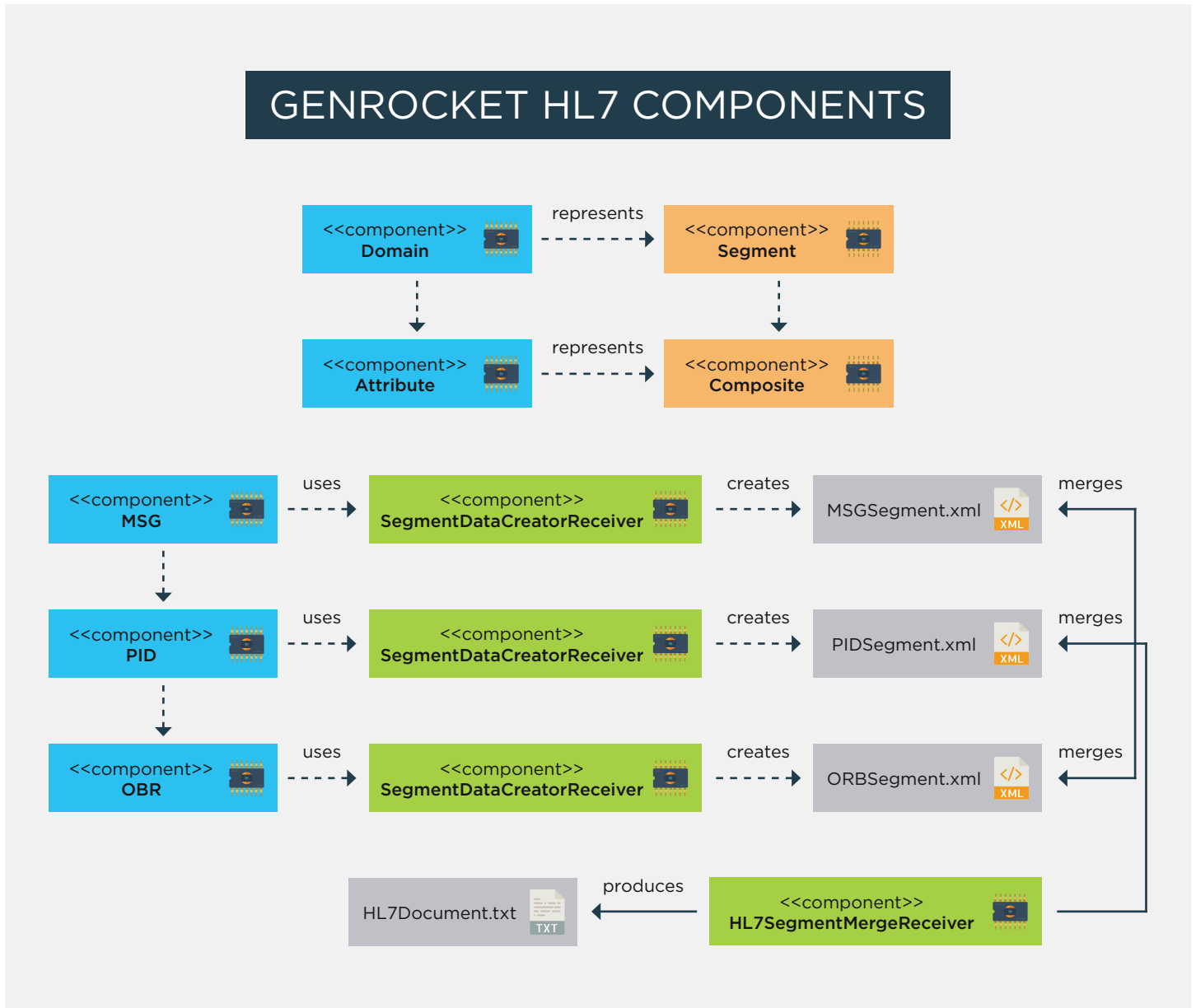
This brief tutorial on the HL7 format illustrates its complexity and the importance of generating test data in a way that faithfully and accurately follows all of its rules.



# GENERATING HL7 TEST DATA WITH GENROCKET

GenRocket uses the concept of a *Domain* to represent an HL7 *Segment* and an *Attribute* represents an HL7 *Composite*. The GenRocket *Receiver* component, in this case the *SegmentDataCreatorReceiver*, morphs a Domain's generated data into name/value pair XML segments. By creating XML segments which represent each HL7 segment, we can ensure full referential integrity between various components. The *HL7SegmentMergeReceiver* can then read these XML segments and merge the segmented data into the required HL7 format.

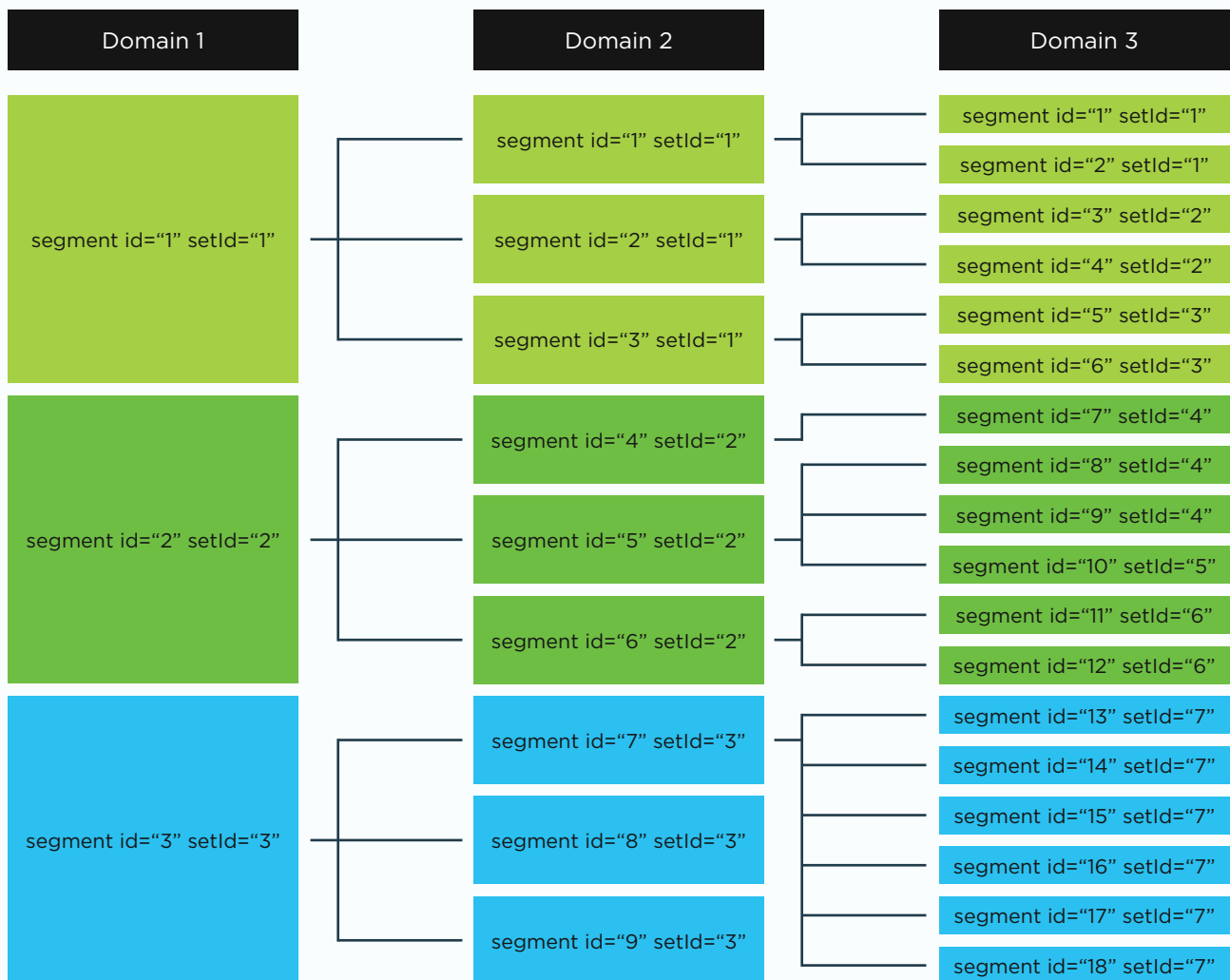
See the diagram below:



The approach leverages the power and versatility of GenRocket *Generators* to create model-based, conditioned test data for any test case and transform the data into an HL7 test data file that conforms to all of the rules and requirements described above.

The GenRocket HL7 solution is completely configurable based on what data segments need to be combined together to meet the file format requirements and the GenRocket Domain model allows the segments to be configured in a logical way that allows the data to be generated quickly and accurately.

## SegmentDataCreatorReceiver Set Based Notation



The diagram above illustrates this versatility. If we start at the top with the first segment of Domain 1 (setid=1) while it shows that Domain 2 may have 3 segments and Domain 3 each has 2 segments.

Note that this is not the case with the second segment of Domain 1 (setid=2). The Domain 2 second segments are the same with 3 segments but then in Domain 3 the segments vary between 1 and 3.

The third segment Domain 1 (setid=3) shows yet another variation. The Domain 2 second segments are the same with 3 segments but then in Domain 3 there are 6 segments defined.

Here is a high-level summary of the GenRocket approach:

- Once a Domain is defined, we associate it to the SegmentDataCreatorReceiver.
- The SegmentDataCreatorReceiver defines the output directory (for segments), the name of the segment, files per directory and records per file.
- A final Domain is created for Segment Merge, using the HL7SegmentMergeReceiver
- The HL7SegmentMergeReceiver merges the segment data into a nested HL7 formatted file.

Following is a sample of the XML definition equivalent and the HL7 output

## The HL7Segment XML Definition Equivalent

```
<HL7segment name="PatientRecord">
  <composites>
    <composite position="1" fieldName="messageType" length="1" example="P"/>
    <composite position="2" fieldName="sequenceNumber" length="5"/>
    <composite position="3" fieldName="patientId" length="25"/>
    <composite position="4" fieldName="accessionNumber" length="9"/>
    <composite position="5" fieldName="socialSecurityNumber" length="9" example="(NNNNNNNNN)"/>
    <composite position="6">
      <composite position="1" fieldName="lastName" length="11"/>
      <composite position="2" fieldName="firstName" length="11"/>
      <composite position="3" fieldName="middleInitial" length="1"/>
    </composite>
    <composite position="7" fieldName="cancelledAssociation" length="1" example="null or "X" - canceled"/>
    <composite position="8">
      <composite position="1" fieldName="patientAge" length="3"/>
      <composite position="2" fieldName="patientDOB" length="8" example="(YYYYMMDD)"/>
    </composite>
    <composite position="9">
      <composite position="1" fieldName="sex" length="1" example="(M, F, or null)"/>
      <composite position="1" fieldName="species" length="1"/>
    </composite>
    <composite position="10" fieldName="dateOfService" length="12" example="YYYYMMDDHHMM"/>
  </composites>
</HL7segment>
```

## Example Output

```
P|1|100000|C3WHWJ8W|001010001|Rivero^Keisha^^^A|Z|77^20170707|F^H|20180620|131
```

## THE OUTCOME

Using GenRocket's Test Data Generation platform, this healthcare company was able to meet all of their test data challenges and realize the following important benefits:

- Generate HL7 files rapidly without the need to access and mask production data
- Generate HL7 files with any level of complexity and complete referential integrity
- Avoid any use of PII with HL7 files that are synthetically generated and 100% secure
- Implement a budget-friendly, cost-effective solution for any testing requirement



If you would like to know more about GenRocket's Test Data Generation platform and our industry solutions, please visit our website at [www.genrocket.com](http://www.genrocket.com).