We plan to replace our CICS screens with Java dialogs. How will the COBOL structures be passed between the Java dialogs and the other COBOL programs?

The isCOBOL Compiler outputs Java classes (actual Java .class files). Each COBOL subprogram becomes a Java class with a method for calling it. This call method accepts the COBOL linkage items as an array of Java Objects. The caller can pass Java Strings or other types of Java Objects that will be converted to the COBOL linkage items using logic similar to a COBOL MOVE statement. Also available is the option for the Java dialogs to use the isCOBOL framework types classes to create the COBOL data-type objects in Java.

Alternatively, a COBOL programmer can use Object-Oriented COBOL (OOCOBOL) syntax to define classes, methods and have precise control over the parameter types, return types and exceptions, to wrap legacy COBOL subprograms so they appear to the Java programmers just as if they were written in the Java programming language, taking full advantage of the object-oriented programming features of Java.

Also, COBOL programs have access to all of the Java classes and types. So it is easy to call Java from COBOL, COBOL from Java, and even extend Java objects using COBOL or COBOL objects using Java.

On Veryant.com, click on Solutions and then Java Integration (or https://www.veryant.com/solutions/cobol-java-integration.html) to read about these options and see actual code samples.

Also the isCOBOL Development System comes with a utility (COBFILEIO) which creates Java classes to allow Java programmers to access COBOL indexed files and records as Java objects. This makes it very natural for a Java programmer to access COBOL data.

Online URL: https://support.veryant.com/phpkb/article.php?id=112