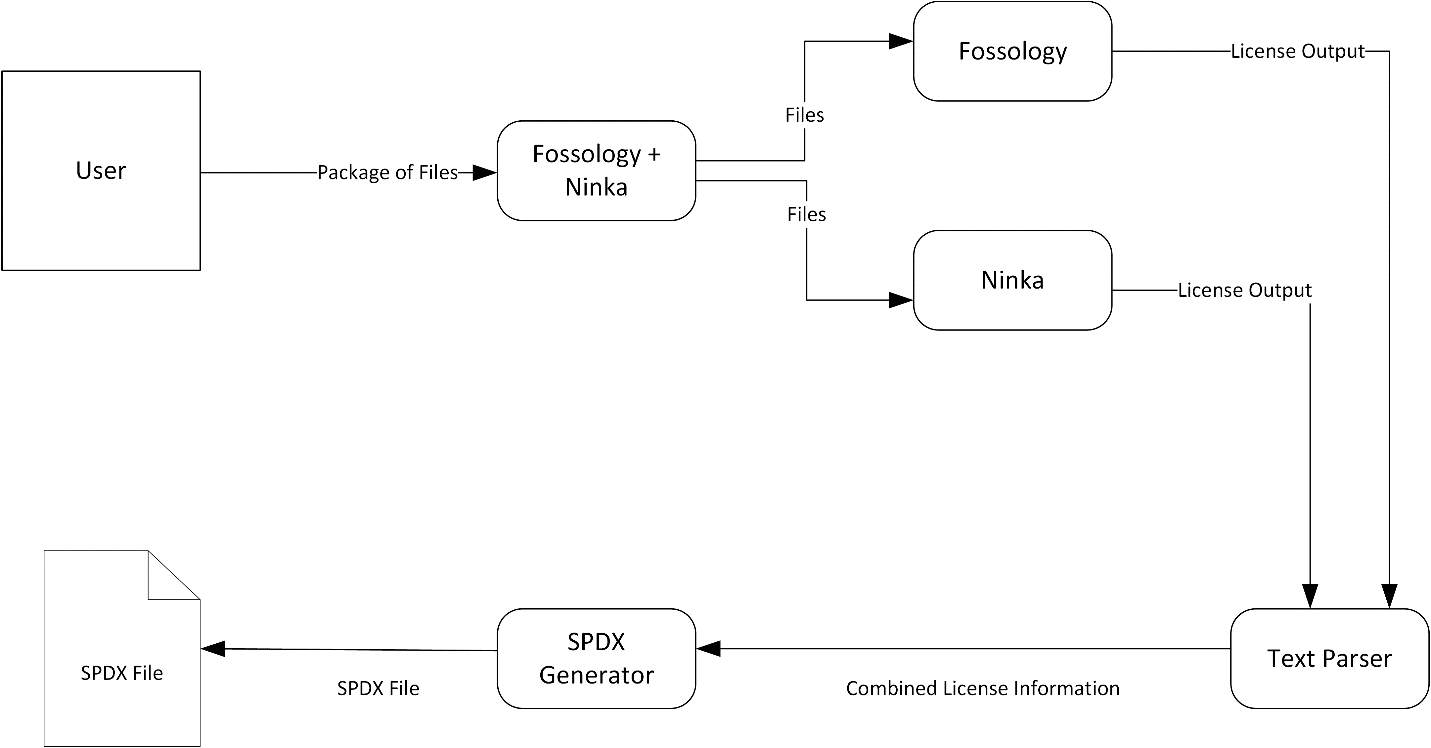
# Data Flow Diagram

****

FOSSOLOY + NINKA

-INPUT: Software #software is a file OR a package

-If INPUT is not package

--FOSSOLOGY(INPUT)

--NINKA(INPUT)

-ELSE

--Unpack(INPUT) as TEMP

--For FILES in TEMP

---FOSSOLOGY(FILE)

---NINKA(FILE)

(FOSSOLOGY and NINKA are external modules, they are treated as black boxes)

TEXT PARSER

-INPUT(FOSSOLGY\_OUT, NINKA\_OUT)

-READ(FOSSOLOGY\_OUT) as R1

-READ(NINKA\_OUT) as R2

-COMPARE(R1, R2) #compares R1 and R2 line by line, assumes the files are sorted in the same order

--IF R1 has license AND R2 has NONE or ERROR

---WRITE license(R1) TO COMBINED\_OUT

--ELSE IF R2 has license and R1 has NONE or ERROR

---WRITE license(R2) TO COMBINED\_OUT

--ELSE IF license(R1) == license(R2)

---WRITE license(R1) TO COMBINED\_OUT #R1 and R2 are identical, either one works

--ELSE IF license(R1) != license(R2) #assuming R1 and R2 have licenses

---WRITE "LICENSE\_DECLARED = NOASSERTION, COMMENTS = CONFLICT" TO COMBINED\_OUT

#not the actual output format, just an example for representation purposes

SPDX GENERATOR

-INPUT(COMBINED\_INFO)

-CREATE FILE FINAL\_DOC, FORMAT = JSON, SCHEMA = SPDX

-IF COMBINED\_INFO.FILE\_FORMAT == PACKAGE

--FOR LINE IN COMBINED INFO #Multiple lines for a package

---WRITE(LINE.LICENSE\_DECLARED) TO FINAL\_DOC.LICENSE\_DECLARED

---WRITE(LINE.COMMENTS) TO FINAL\_DOC.COMMENTS #Even if no conflict, it can check for other comments that may be put in later

-ELSE

--WRITE(LINE.LICENSE\_DECLARED) TO FINAL\_DOC.LICENSE\_FOUND

--WRITE(LINE.COMMENTS) TO FINAL\_DOC.COMMENTS