What is claimed is:

1. A method for software specification translation, the method including: receiving a first software specification specified in a first programming language, wherein the first programming language is a procedural programming language; receiving a second software specification specified in a second programming language, wherein the second programming language is a procedural programming language; receiving a third software specification specified in a third programming language different from the first and second programming languages, the third software specification referencing the first and second software specifications and constraining a flow of execution, and the third software specification defining one or more data relationships between the first software specification and the second software specification, where the data relationships are defined based on references to specified datasets; forming a representation of the first software specification in a fourth programming language different from the first, second, and third programming languages, wherein the fourth programming language is a dataflow graph-based programming language; forming a representation of the second software specification in the fourth programming language; analyzing the third software specification to identify the one or more data relationships; and forming a combined representation of the first software specification and the second software specification in the fourth programming language including forming connections in the fourth programming language between the representation of the first software specification in the fourth programming language and the representation of the second software specification in the fourth programming language according to the identified one or more data relationships.

2. The method of claim 1 wherein the fourth programming language enables parallelism between different portions of a software specification.

3. The method of claim 2 wherein the fourth programming language enables a plurality of types of parallelism including: a first type of parallelism enabling multiple instances of a portion of a software specification to operate on different portions of an input data stream; and a second type of parallelism enabling different portions of a software specification to execute concurrently on different portions of the input data stream.

4. The method of claim 1 wherein the second programming language is the same as the first programming language.

5. The method of claim 1 wherein the one or more data relationships between the first software specification and the second software specification include at least one data relationship that corresponds to the first software specification receiving data from a first dataset and the second software specification providing data to the first dataset.

6. The method of claim 1 wherein the connections in the fourth programming language correspond to directed links representing flows of data.

7. The method of claim 1 wherein the first software specification is configured to interact with one or more datasets, each dataset having an associated dataset type of a plurality of dataset types in the first software specification, and the second software specification is configured to interact with one or more datasets, each data set having an associated type of the plurality of dataset types in the second software specification, the method further including: processing the first software specification, the processing including: identifying the one or more datasets of the first software specification, and for each of the identified one or more datasets, determining the associated type of the dataset in the first software specification; and forming a representation of the first software specification in the fourth programming language, including, for each of the identified one or more datasets, forming a specification of the dataset in the fourth programming language, the specification of the dataset in the fourth programming language having a type corresponding to the associated type of the dataset in the first programming language; wherein at least one of the specifications of the one or more datasets in the fourth programming language has: an input dataset type or an output dataset type; processing the second software specification, the processing including: identifying the one or more datasets of the second software specification and for each of the identified one or more datasets, determining the associated type of the dataset in the second software specification; and forming a representation of the second software specification in the fourth programming language, including, for each of the identified one or more datasets, forming a specification of the dataset in the fourth programming language, the specification of the dataset in the fourth programming language having a type corresponding to the associated type of the dataset in the first programming language; wherein at least one of the specification of the one or more datasets in the fourth programming language enables: an input function or an output function.

8. The method of claim 7 wherein forming the combined representation includes at least one of: forming one or more connections to replace connections between the specifications of the one or more datasets of the second software specification in the fourth programming language enabling input functions and the representation of the second software specification in the fourth programming language with connections between the representation of the first software specification in the fourth programming language and the representation of the second software specification in the fourth programming language; or forming one or more connections to replace connections between the specification of the one or more datasets of the first software specification in the fourth programming language enabling input functions and the representation of the first software specification in the fourth programming language with connections between the representation of the second software specification in the fourth programming language and the representation of the first software specification in the fourth programming language.

9. The method of claim 8 further including: preserving the one or more datasets of the first software specification in the fourth programming language enabling output functions in the representation of the first software specification in the fourth programming language, or preserving the one or more datasets of the second software specification in the fourth programming language enabling output functions in the representation of the second software specification in the fourth programming language.

10. The method of claim 7 wherein: the first software specification includes one or more data transformation operations and analyzing the first software specification includes identifying at least some of the one or more data transformation operations and classifying the identified data transformation operations into a corresponding data transformation type of the fourth programming language, and forming the representation of the first software specification in the fourth programming language includes, for each of the identified data transformation operations, forming a specification of the data transformation operation in the fourth programming language, the specification of the data transformation operation in the fourth programming language enabling a data transform operation corresponding to the data transformation type of the identified data transformation operation in the first programming language.

11. The method of claim 7 wherein at least one of the specifications of the one or more datasets in the fourth programming language has a read-only random access dataset type.

12. The method of claim 7 wherein determining the associated type of the dataset in the first software specification includes analyzing parameters of dataset definitions and commands that access the dataset.

13. The method of claim 12 wherein the parameters include one or more of a file organization associated with the dataset, an access mode associated with the dataset, a mode used to open the dataset, and input-output operations.

14. The method of claim 1 further including storing the combined representation of the first software specification and the second software specification in a storage medium.

15. The method of claim 1 wherein the first software specification defines one or more data processing operations that interact with one or more datasets, and the second software specification defines one or more data processing operations that interact with one or more datasets.

16. The method of claim 15 wherein the third software specification defines one or more data relationships between the one or more datasets of the first software specification and the one or more datasets of the second software specification.

17. The method of claim 1, wherein forming the combined representation includes forming at least one of the connections in the fourth programming language to replace a connection between: (1) a specification of at least one dataset of the first software specification in the fourth programming language enabling input functions and (2) a representation of the second software specification in the fourth programming language.

18. Software stored in a non-transitory form on a computer-readable medium, for software specification translation, the software including instructions for causing a computing system to:

receive a first software specification specified in a first programming language, wherein the first programming language is a procedural programming language; receive a second software specification specified in a second programming language, wherein the second programming language is a procedural programming language; receive a third software specification specified in a third programming language different from the first and second programming languages, the third software specification referencing the first and second software specifications and constraining a flow of execution, and the third software specification defining one or more data relationships between the first software specification and the second software specification, where the data relationships are defined based on references to specified datasets; form a representation of the first software specification in a fourth programming language different from the first, second, and third programming languages, wherein the fourth programming language is a dataflow graph-based programming language; form a representation of the second software specification in the fourth programming language; analyze the third software specification to identify the one or more data relationships; and form a combined representation of the first software specification and the second software specification in the fourth programming language including forming connections in the fourth programming language between the representation of the first software specification in the fourth programming language and the representation of the second software specification in the fourth programming language according to the identified one or more data relationships.

19. The software of claim 18 wherein the one or more data relationships between the first software specification and the second software specification include at least one data relationship that corresponds to the first software specification receiving data from a first dataset and the second software specification providing data to the first dataset.

20. The software of claim 18 wherein the connections in the fourth programming language correspond to directed links representing flows of data.

21. The software of claim 18 wherein the first software specification is configured to interact with one or more datasets, each dataset having an associated dataset type of a plurality of dataset types in the first software specification, and the second software specification is configured to interact with one or more datasets, each data set having an associated type of the plurality of dataset types in the second software specification, the software further including instructions for causing a computing system to: process the first software specification, the processing including:

identifying the one or more datasets of the first software specification, and for each of the identified one or more datasets, determining the associated type of the dataset in the first software specification; and forming a representation of the first software specification in the fourth programming language, including, for each of the identified one or more datasets, forming a specification of the dataset in the fourth programming language, the specification of the dataset in the fourth programming language having a type corresponding to the associated type of the dataset in the first programming language; wherein at least one of the specifications of the one or more datasets in the fourth programming language has: an input dataset type or an output dataset type; process the second software specification, the processing including: identifying the one or more datasets of the second software specification and for each of the identified one or more datasets, determining the associated type of the dataset in the second software specification; and forming a representation of the second software specification in the fourth programming language, including, for each of the identified one or more datasets, forming a specification of the dataset in the fourth programming language, the specification of the dataset in the fourth programming language having a type corresponding to the associated type of the dataset in the first programming language; wherein at least one of the specification of the one or more datasets in the fourth programming language enables: an input function or an output function.

22. The software of claim 21 wherein forming the combined representation includes at least one of: forming one or more connections to replace connections between the specifications of the one or more datasets of the second software specification in the fourth programming language enabling input functions and the representation of the second software specification in the fourth programming language with connections between the representation of the first software specification in the fourth programming language and the representation of the second software specification in the fourth programming language; or forming one or more connections to replace connections between the specification of the one or more datasets of the first software specification in the fourth programming language enabling input functions and the representation of the first software specification in the fourth programming language with connections between the representation of the second software specification in the fourth programming language and the representation of the first software specification in the fourth programming language.

23. The software of claim 22 further including instructions for causing a computing system to: preserve the one or more datasets of the first software specification in the fourth programming language enabling output functions in the representation of the first software specification in the fourth programming language, or preserve the one or more datasets of the

second software specification in the fourth programming language enabling output functions in the representation of the second software specification in the fourth programming language.

24. The software of claim 21 wherein: the first software specification includes one or more data transformation operations and analyzing the first software specification includes identifying at least some of the one or more data transformation operations and classifying the identified data transformation operations into a corresponding data transformation type of the fourth programming language, and forming the representation of the first software specification in the fourth programming language includes, for each of the identified data transformation operations, forming a specification of the data transformation operation in the fourth programming language, the specification of the data transformation operation in the fourth programming language enabling a data transform operation corresponding to the data transformation type of the identified data transformation operation in the first programming language.

25. The software of claim 21 wherein at least one of the specifications of the one or more datasets in the fourth programming language has a read-only random access dataset type.

26. The software of claim 21 wherein determining the associated type of the dataset in the first software specification includes analyzing parameters of dataset definitions and commands that access the dataset.

27. The software of claim 26 wherein the parameters include one or more of a file organization associated with the dataset, an access mode associated with the dataset, a mode used to open the dataset, and input-output operations.

28. The software of claim 18 further including instructions for causing a computing system to store the combined representation of the first software specification and the second software specification in a storage medium.

29. The software of claim 18 wherein the first software specification defines one or more

data processing operations that interact with one or more datasets, and the second software specification defines one or more data processing operations that interact with one or more datasets.

30. The software of claim 29 wherein the third software specification defines one or more data relationships between the one or more datasets of the first software specification and the one or more datasets of the second software specification.

31. The software of claim 18, wherein forming the combined representation includes forming at least one of the connections in the fourth programming language to replace a connection between: (1) a specification of at least one dataset of the first software specification in the fourth programming language enabling input functions and (2) a representation of the second software specification in the fourth programming language.

32. A computing system for software specification translation, the computing system including: an input device or port configured to receive software specifications, the software specifications including: a first software specification specified in a first programming language, wherein the first programming language is a procedural programming language; a second software specification specified in a second programming language, wherein the second programming language is a procedural programming language; a third software specification specified in a third programming language different from the first and second programming languages, the third software specification referencing the first and second software specifications and constraining a flow of execution, and the third software specification defining one or more data relationships between the first software specification and the second software specification, where the data relationships are defined based on references to specified datasets; and at least one processor configured to process the received software specifications, the processing including: forming a representation of the first software specification in a fourth programming language different from the first, second, and third programming languages, wherein the fourth programming language is a dataflow graph-based programming language; forming a representation of the second software specification in the fourth programming language; analyzing the third software specification to identify the one or more data relationships; and forming a combined representation of the first software specification and the second software specification in the fourth programming language including forming connections in the fourth programming language between the representation of the first software

specification in the fourth programming language and the representation of the second software specification in the fourth programming language according to the identified one or more data relationships.

33. The computing system of claim 32 wherein the fourth programming language enables parallelism between different portions of a software specification.

34. The computing system of claim 33 wherein the fourth programming language enables a plurality of types of parallelism including: a first type of parallelism enabling multiple instances of a portion of a software specification to operate on different portions of an input data stream; and a second type of parallelism enabling different portions of a software specification to execute concurrently on different portions of the input data stream.

35. The computing system of claim 32 wherein the second programming language is the same as the first programming language.

36. The computing system of claim 32 wherein the one or more data relationships between the first software specification and the second software specification include at least one data relationship that corresponds to the first software specification receiving data from a first dataset and the second software specification providing data to the first dataset.

37. The computing system of claim 32 wherein the connections in the fourth programming language correspond to directed links representing flows of data.

38. The computing system of claim 32 wherein the first software specification is configured to interact with one or more datasets, each dataset having an associated dataset type of a plurality of dataset types in the first software specification, and the second software specification is configured to interact with one or more datasets, each data set having an associated type of the plurality of dataset types in the second software specification, the computing system further configured to: process the first software specification, the processing including: identifying the

one or more datasets of the first software specification, and for each of the identified one or more datasets, determining the associated type of the dataset in the first software specification; and forming a representation of the first software specification in the fourth programming language, including, for each of the identified one or more datasets, forming a specification of the dataset in the fourth programming language, the specification of the dataset in the fourth programming language having a type corresponding to the associated type of the dataset in the first programming language; wherein at least one of the specifications of the one or more datasets in the fourth programming language has: an input dataset type or an output dataset type; process the second software specification, the processing including: identifying the one or more datasets of the second software specification and for each of the identified one or more datasets, determining the associated type of the dataset in the second software specification; and forming a representation of the second software specification in the fourth programming language, including, for each of the identified one or more datasets, forming a specification of the dataset in the fourth programming language, the specification of the dataset in the fourth programming language having a type corresponding to the associated type of the dataset in the first programming language; wherein at least one of the specification of the one or more datasets in the fourth programming language enables: an input function or an output function.

39. The computing system of claim 38 wherein forming the combined representation includes at least one of: forming one or more connections to replace connections between the specifications of the one or more datasets of the second software specification in the fourth programming language enabling input functions and the representation of the second software specification in the fourth programming language with connections between the representation of the first software specification in the fourth programming language and the representation of the second software specification in the fourth programming language; or forming one or more connections to replace connections between the specification of the one or more datasets of the first software specification in the fourth programming language enabling input functions and the representation of the first software specification in the fourth programming language with connections between the representation of the second software specification in the fourth programming language and the representation of the first software specification in the fourth programming language.

40. The computing system of claim 39 further configured to: preserve the one or more datasets of the first software specification in the fourth programming language enabling output functions in the representation of the first software specification in the fourth programming language, or preserve the one or more datasets of the second software specification in the fourth

programming language enabling output functions in the representation of the second software specification in the fourth programming language.

41. The computing system of claim 38 wherein: the first software specification includes one or more data transformation operations and analyzing the first software specification includes identifying at least some of the one or more data transformation operations and classifying the identified data transformation operations into a corresponding data transformation type of the fourth programming language, and forming the representation of the first software specification in the fourth programming language includes, for each of the identified data transformation operations, forming a specification of the data transformation operation in the fourth programming language, the specification of the data transformation operation in the fourth programming language enabling a data transform operation corresponding to the data transformation type of the identified data transformation operation in the first programming language.

42. The computing system of claim 38 wherein at least one of the specifications of the one or more datasets in the fourth programming language has a read-only random access dataset type.

43. The computing system of claim 38 wherein determining the associated type of the dataset in the first software specification includes analyzing parameters of dataset definitions and commands that access the dataset.

44. The computing system of claim 43 wherein the parameters include one or more of a file organization associated with the dataset, an access mode associated with the dataset, a mode used to open the dataset, and input-output operations.

45. The computing system of claim 32 further configured to store the combined representation of the first software specification and the second software specification in a storage medium.

46. The computing system of claim 32 wherein the first software specification defines one or more data processing operations that interact with one or more datasets, and the second software specification defines one or more data processing operations that interact with one or more datasets.

47. The computing system of claim 46 wherein the third software specification defines one or more data relationships between the one or more datasets of the first software specification and the one or more datasets of the second software specification.

48. The computing system of claim 32, wherein forming the combined representation includes forming at least one of the connections in the fourth programming language to replace a connection between: (1) a specification of at least one dataset of the first software specification in the fourth programming language enabling input functions and (2) a representation of the second software specification in the fourth programming language.

49. A computing system for software specification translation, the computing system including: means for receiving software specifications, the software specifications including: a first software specification specified in a first programming language, wherein the first programming language is a procedural programming language; a second software specification specified in a second programming language, wherein the second programming language is a procedural programming language; a third software specification specified in a third programming language different from the first and second programming languages, the third software specification referencing the first and second software specifications and constraining a flow of execution, and the third software specification defining one or more data relationships between the first software specification and the second software specification, where the data relationships are defined based on references to specified datasets; and means for processing the received software specifications, the processing including: forming a representation of the first software specification in a fourth programming language different from the first, second, and third programming languages, wherein the fourth programming language is a dataflow graph-based programming language; forming a representation of the second software specification in the fourth programming language; analyzing the third software specification to identify the one or more data relationships; and forming a combined representation of the first software specification and the second software specification in the fourth programming language including forming connections in the fourth programming language between the representation

of the first software specification in the fourth programming language and the representation of the second software specification in the fourth programming language according to the identified one or more data relationships.

50. The software of claim 49 wherein the fourth programming language enables parallelism between different portions of a software specification.

51. The software of claim 50 wherein the fourth programming language enables a plurality of types of parallelism including: a first type of parallelism enabling multiple instances of a portion of a software specification to operate on different portions of an input data stream; and a second type of parallelism enabling different portions of a software specification to execute concurrently on different portions of the input data stream.

52. The software of claim 49 wherein the second programming language is the same as the first programming language.