**CLAIMS**

1. A system, comprising:

a memory that stores computer executable components; and

a processor that executes the computer executable components stored in the memory, wherein the computer executable components comprise:

a model component that learns cluster assignments of classes in a monolithic application based on runtime traces of executed test cases, wherein the runtime traces are appended with labels comprising respective business contexts associated with the runtime traces; and

a cluster component that employs the model component to generate clusters of the classes based on the cluster assignments to identify one or more microservices of the monolithic application.

2. The system of claim 1, wherein the computer executable components comprise:

a refinement component that refines the one or more microservices based on at least one of data dependency of the monolithic application or a static call graph of the monolithic application.

3. The system of claim 1, wherein the computer executable components comprise:

a second model component that generates one or more causal graphs based on the runtime traces of the executed test cases to capture at least one of first order temporal dependencies or high order temporal dependencies of at least one of the monolithic application or the cluster assignments of the classes in the monolithic application.

4. The system of claim 1, wherein the model component learns at least one of the cluster assignments or graph embeddings of the classes in the monolithic application using causal sequences of one or more causal graphs generated based on the runtime traces of the executed test cases.

5. The system of claim 1, wherein the executed test cases comprise business function test cases that provide business functionalities of the monolithic application in the runtime traces, thereby facilitating at least one of improved performance, improved efficiency, or reduced computational costs of at least one of the processor or the system in executing application modernization of the monolithic application.

6. A computer-implemented method, comprising:

training, by a system operatively coupled to a processor, a model to learn cluster assignments of classes in a monolithic application based on runtime traces of executed test cases, wherein the runtime

traces are appended with labels comprising respective business contexts associated with the runtime traces; and

employing, by the system, the model to generate clusters of the classes based on the cluster assignments to identify one or more micro services of the monolithic application.

7. The computer-implemented method of claim 6, further comprising:

refining, by the system, the one or more microservices based on at least one of data dependency of the monolithic application or a static call graph of the monolithic application.

8. The computer-implemented method of claim 6, further comprising:

generating, by the system, one or more causal graphs based on the runtime traces of the executed test cases to capture at least one of first order temporal dependencies or high order temporal dependencies of at least one of the monolithic application or the cluster assignments of the classes in the monolithic application.

9. The computer-implemented method of claim 6, wherein the training comprises:

training, by the system, the model to learn at least one of the cluster assignments or graph embeddings of the classes in the monolithic application using causal sequences of one or more causal graphs generated based on the runtime traces of the executed test cases.

10. The computer-implemented method of claim 6, wherein the executed test cases comprise business function test cases that provide business functionalities of the monolithic application in the runtime traces, thereby facilitating at least one of improved performance, improved efficiency, or reduced computational costs of at least one of the processor or the system in executing application modernization of the monolithic application.

11. A computer program product facilitating a process to generate microservices from a monolithic application based on runtime traces, the computer program product comprising a computer readable storage medium having program instructions embodied therewith, the program instructions executable by a processor to cause the processor to:

train, by the processor, a model to learn cluster assignments of classes in the monolithic application based on runtime traces of executed test cases, wherein the runtime traces are appended with labels comprising respective business contexts associated with the runtime traces; and

employ, by the processor, the model to generate clusters of the classes based on the cluster assignments to identify one or more micro services of the monolithic application.

12. The computer program product of claim 11, wherein the program instructions are further executable by the processor to cause the processor to:

refine, by the processor, the one or more microservices based on at least one of data dependency of the monolithic application or a static call graph of the monolithic application.

13. The computer program product of claim 11, wherein the program instructions are further executable by the processor to cause the processor to:

generate, by the processor, a causal graph based on the runtime traces of the executed test cases to capture at least one of first order temporal dependencies or high order temporal dependencies of at least one of the monolithic application or the cluster assignments of the classes in the monolithic application.

14. The computer program product of claim 11, wherein the program instructions are further executable by the processor to cause the processor to:

train, by the processor, the model to learn at least one of the cluster assignments or graph embeddings of the classes in the monolithic application using causal sequences of one or more causal graphs generated based on the runtime traces of the executed test cases.

15. The computer program product of claim 11, wherein the executed test cases comprise business function test cases that provide business functionalities of the monolithic application in the runtime traces.

16. A system, comprising:

a memory that stores computer executable components; and

a processor that executes the computer executable components stored in the memory, wherein the computer executable components comprise:

a collection component that:

collects runtime traces of test cases executed on a monolithic application, and

appends labels to the runtime traces comprising respective business contexts associated with the runtime traces;

a model component that learns cluster assignments of classes in the monolithic application based on the runtime traces and the respective business contexts; and

a cluster component that employs the model component to generate clusters of the classes based on the cluster assignments to identify one or more microservices of the monolithic application.

17. The system of claim 16, wherein the computer executable components comprise:

a refinement component that refines the one or more microservices based on at least one of data dependency of the monolithic application or a static call graph of the monolithic application.

18. The system of claim 16, wherein the computer executable components comprise:

a second model component that generates one or more causal graphs based on the runtime traces to capture at least one of first order temporal dependencies or high order temporal dependencies of at least one of the monolithic application or the cluster assignments of the classes in the monolithic application.

19. The system of claim 16, wherein the model component learns at least one of the cluster assignments or graph embeddings of the classes in the monolithic application using causal sequences of one or more causal graphs generated based on the runtime traces.

20. The system of claim 16, wherein the test cases comprise business function test cases that provide business functionalities of the monolithic application in the runtime traces, thereby facilitating at least one of improved performance, improved efficiency, or reduced computational costs of at least one of the processor or the system in executing application modernization of the monolithic application.

21. A computer-implemented method, comprising:

collecting, by a system operatively coupled to a processor, runtime traces of test cases executed on a monolithic application;

appending, by the system, labels to the runtime traces comprising respective business contexts associated with the runtime traces;

training, by the system, a model to learn cluster assignments of classes in the monolithic application based on the runtime traces and the respective business contexts; and

employing, by the system, the model to generate clusters of the classes based on the cluster assignments to identify one or more micro services of the monolithic application.

22. The computer-implemented method of claim 21, further comprising:

refining, by the system, the one or more microservices based on at least one of data dependency of the monolithic application or a static call graph of the monolithic application.

23. The computer-implemented method of claim 21, further comprising:

generating, by the system, one or more causal graphs based on the runtime traces to capture at least one of first order temporal dependencies or high order temporal dependencies of at least one of the monolithic application or the cluster assignments of the classes in the monolithic application.

24. The computer-implemented method of claim 21, wherein the training comprises:

training, by the system, the model to learn at least one of the cluster assignments or graph embeddings of the classes in the monolithic application using causal sequences of one or more causal graphs generated based on the runtime traces.

25. The computer-implemented method of claim 21, wherein the test cases comprise business function test cases that provide business functionalities of the monolithic application in the runtime traces.