ON THE USE OF DATABASE FRAMEWORKS IN JAVA PROJECTS

A LARGE-SCALE HISTORICAL EMPIRICAL ANALYSIS

Mathieu Goeminne
mathieu.goeminne@umons.ac.be

Tom Mens
tom.mens@umons.ac.be

Alexandre Decan
alexandre.decan@umons.ac.be

UMONS Université de Mons

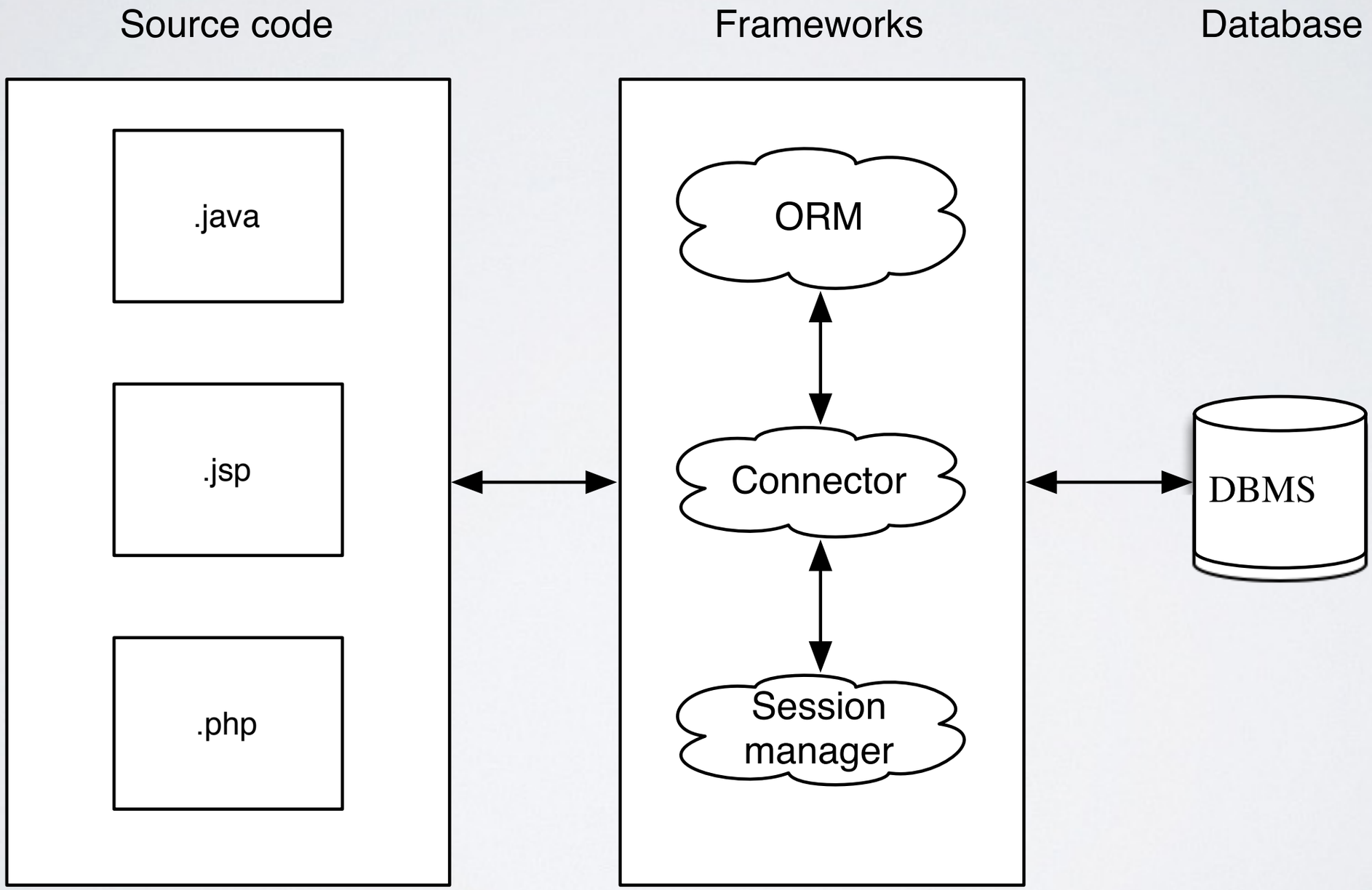
Software Engineering Lab

Faculté des Sciences

GENERAL CONTEXT

- FNRS interuniversity research collaboration with the University of Namur.
- Study of software systems that use a database.
- Understanding the dynamics, issues and evolution of such systems.
- Need to take into account their specificities:
  - How is their connection managed?
  - How do the source code and the database co-evolve?
  - How can we assess and increase the quality of such systems?

DATABASE FRAMEWORKS?



ON THIS PRESENTATION

- Focus on database frameworks.
- Empirical study for answering 3 research questions:
  - **Q1:** Which combinations of DB frameworks are being used simultaneously?
  - **Q2:** How long do DB frameworks 'survive' in the projects that use them?
  - **Q3:** Does the introduction of a DB framework influence the survivability of another one?

CASE STUDY

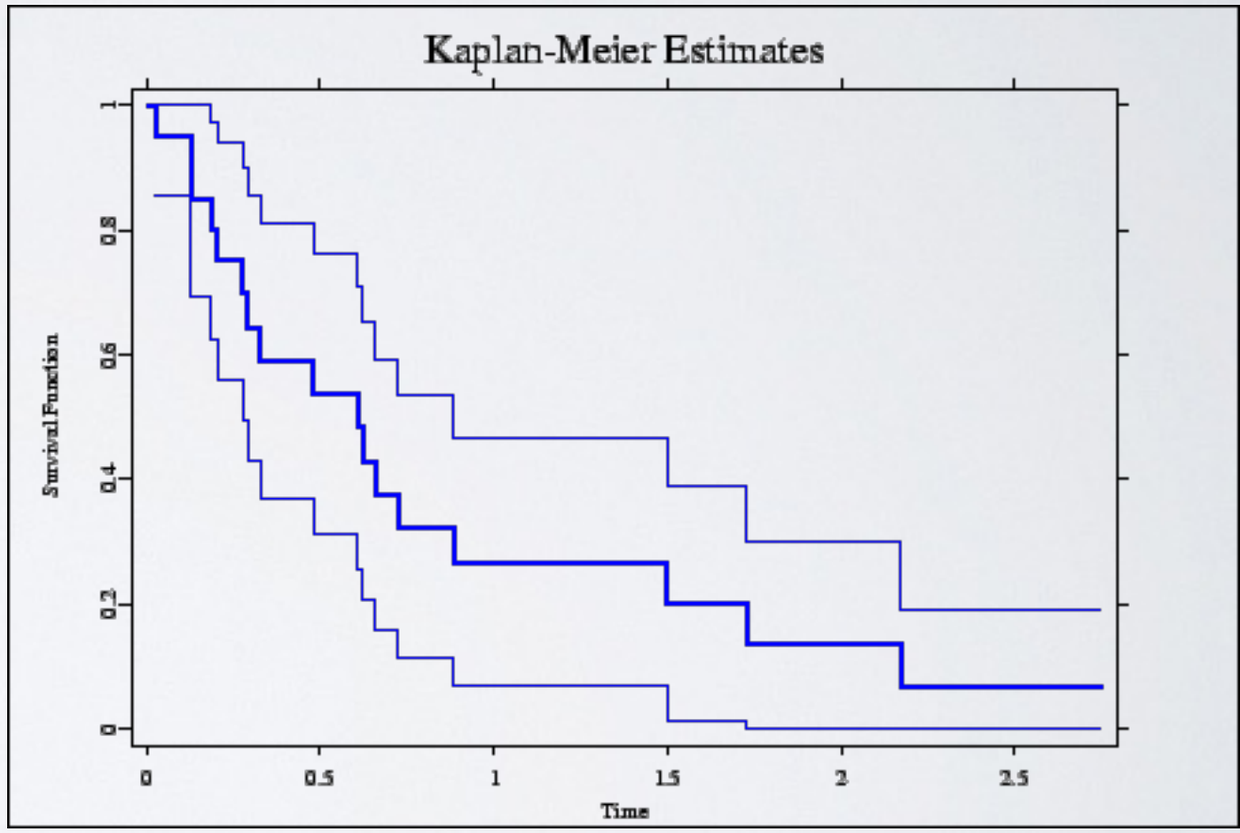
- 20 Java DB frameworks.
- Top **5** retained: JDBC, Spring, JPA, Vaadin, Hibernate.
- 13,307 Java projects from the Github Java Corpus (MSR10): **3,707** of them use at least one retained framework)

SURVIVAL ANALYSIS: KAPLAN-MEIER ESTIMATOR

- Estimates the **survival function** of a population

- Takes into account **right censoring** (the event may occur after the last observation)

- **Mantel-Haenszel test** for determining if 2 survival functions differ.



Q1 - Which combinations of DB frameworks are being used simultaneously?

- In most cases (98%), different frameworks used in a project are used in combination.
- JDBC is used as the single DB framework in 56% of all projects.
- Most projects (83%) using Hibernate also use JDBC. Hibernate is almost never (3%) used alone.
- JDBC used as support for the other FW, providing unique services?

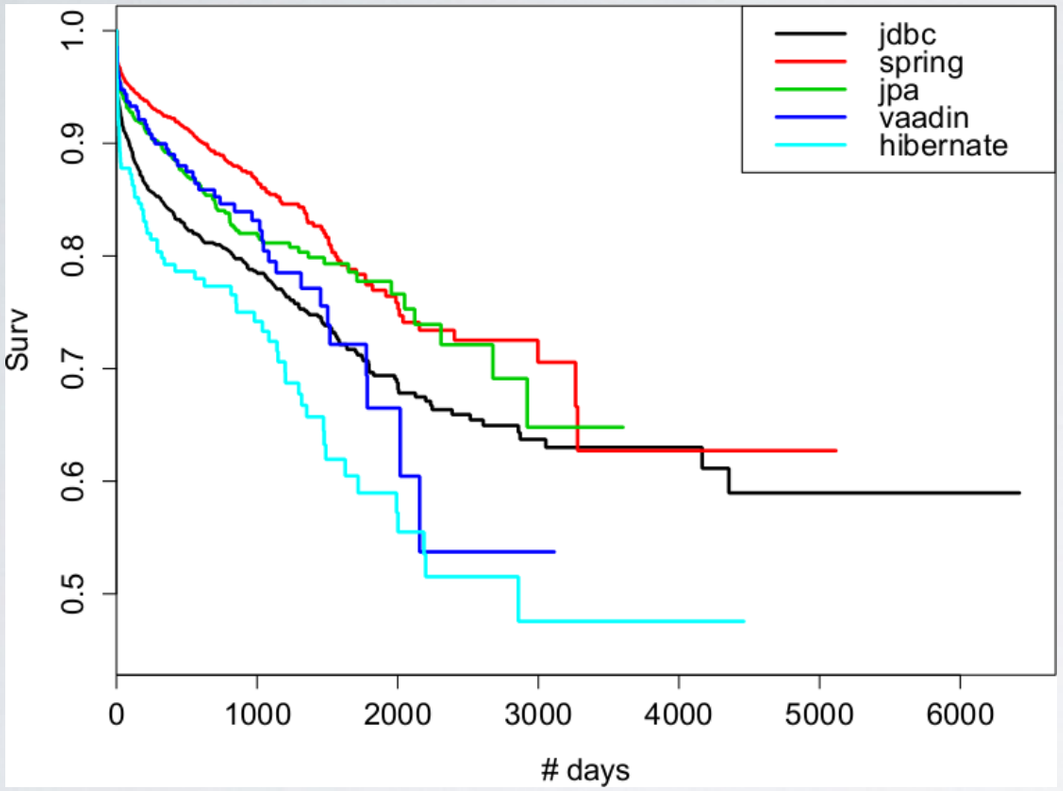
# simultaneous fw. → ↓ total # frameworks used	1	2	3	4	5
1	2,443				
2	22	776			
3	2	16	328		
4	0	0	18	104	
5	0	0	1	1	5

projects using X fw vs Y fw simultaneously

	Spring	JPA	Vaadin	Hibernate
JDBC	645	565	143	192
Spring		558	76	156
JPA			98	105
Vaadin				22

projects using X and Y simultaneously

Q2 - How long do database frameworks 'survive' in the projects?



Survival functions of top 5 FW according to Kaplan-Meier estimator

$A \rightarrow$ $\downarrow B$	Spring	JPA	Vaadin	Hibernate
JDBC	< **0.001** [-]	**0.001** [-]	0.242	0.010
Spring	—	0.030	0.017	< **0.001** [+]
JPA		—	0.427	< **0.001** [+]
Vaadin			—	0.017

p-values for difference tests of survival functions

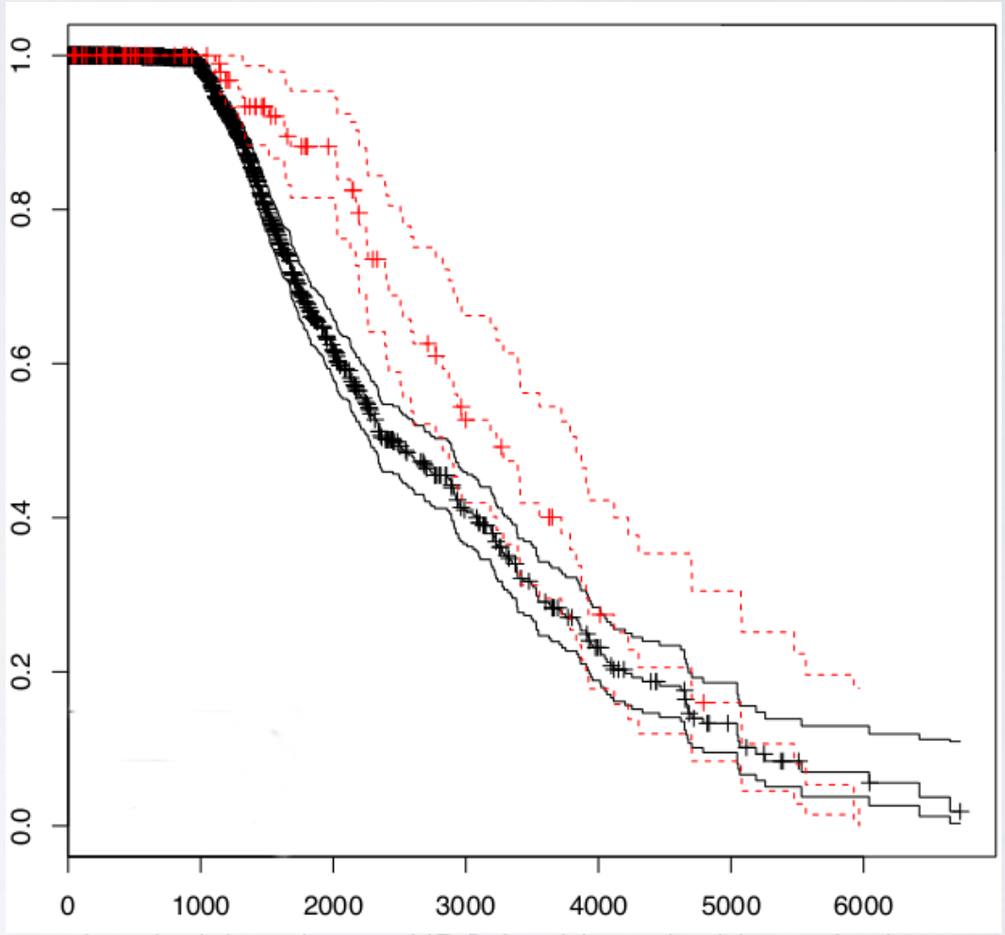
- If a framework is introduced, it remains in >45% of all the projects.
- Different trends. After 30 days:
  - Hibernate disappears from 11.7% of projects
  - Spring disappears from 3.7% of projects
- Partial order: JPA and Spring have higher survival rates than JDBC and Hibernate.
  - No clear relation between the order and the *popularity* of the considered frameworks.

Q3 - Does the introduction of a database framework influence the survivability of another one?

- Visually, the introduction of some frameworks seems to have a favorable influence.

- After Bonferroni correction, **no significant difference in the survival functions**… at project level.

- In the future: file level analyse.



Survival functions of JDBC with and without Spring

Introduced → ↓ Tested	JDBC	Spring	JPA	Vaadin	Hibernate
JDBC	—	**0.045**$^{+}$	**0.015**$^{+}$	0.198	**0.016**$^{+}$
Spring	0.300	—	**0.021**$^{+}$	0.678	**0.006**$^{+}$
JPA	**0.020**$^{?}$	0.972	—	0.339	0.094
Vaadin	0.507	0.845	0.299	—	0.462
Hibernate	0.110	0.835	0.913	0.921	—

CONCLUSIONS

- **Important coexistence**
  - Especially between JDBC and the other considered DB frameworks.
  - **JDBC is still often present in projects** despite the use frameworks providing more advanced, higher level services.
- In our study, there is **no evidence for competition** between frameworks.
  - An introduced framework has **an ≳45% of chance that it will remain**.
  - The disappearance of a framework doesn't seem to be related to the introduction of an other one.

FUTURE WORK

- **Finer granularity:** from projects to packages, files, methods, …
- Analysis of (un)**frequently used framework services**.
- Looking for evolutionary patterns:
  - New frameworks introduced for their unique services?
  - Simple SQL queries are replaced by framework services, while complex queries remain as it?
- Extension to frameworks for **less structured databases**, NoSQL, etc.