Using Microservices for Legacy Software Modernization

Holger Knoche
Wilhelm Hasselbring
Software Engineering Group, Kiel University
http://se.informatik.uni-kiel.de/

February 21th, 2019

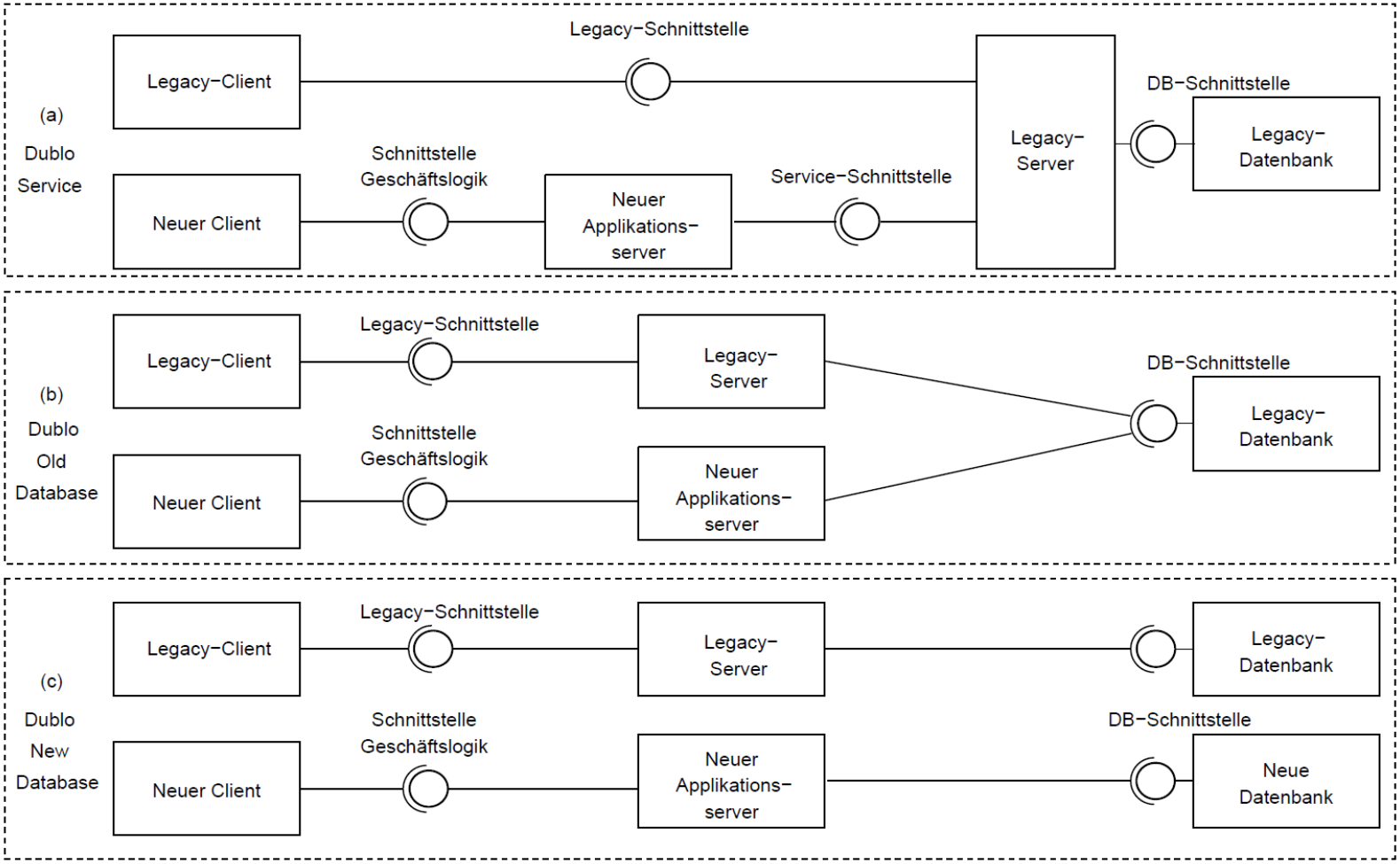
Journal-first presentation: Knoche, H. and Hasselbring, W. (2018)
“Using Microservices for Legacy Software Modernization”
IEEE Software, 35 (3). pp. 44-49. DOI 10.1109/MS.2018.2141035

Christian-Albrechts-Universität zu Kiel

Agenda

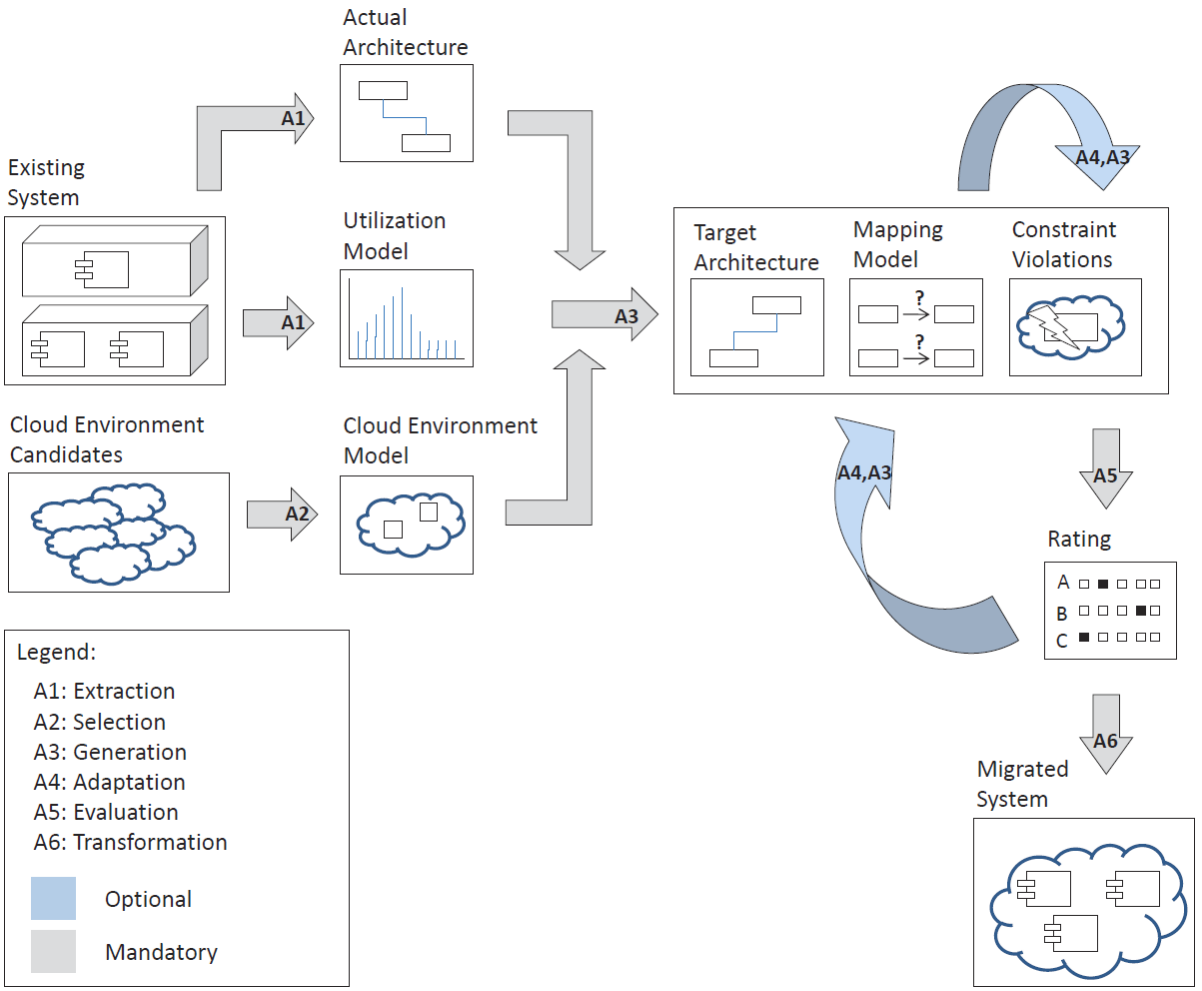
1. Software Migration and Modernization
2. Motivation for Migrating to Microservices
3. Our Migration Case Study
4. Summary, so far

Migration to SOA



Dublo Migration Pattern [Hasselbring et al. 2004, 2008]

Migration to the Cloud

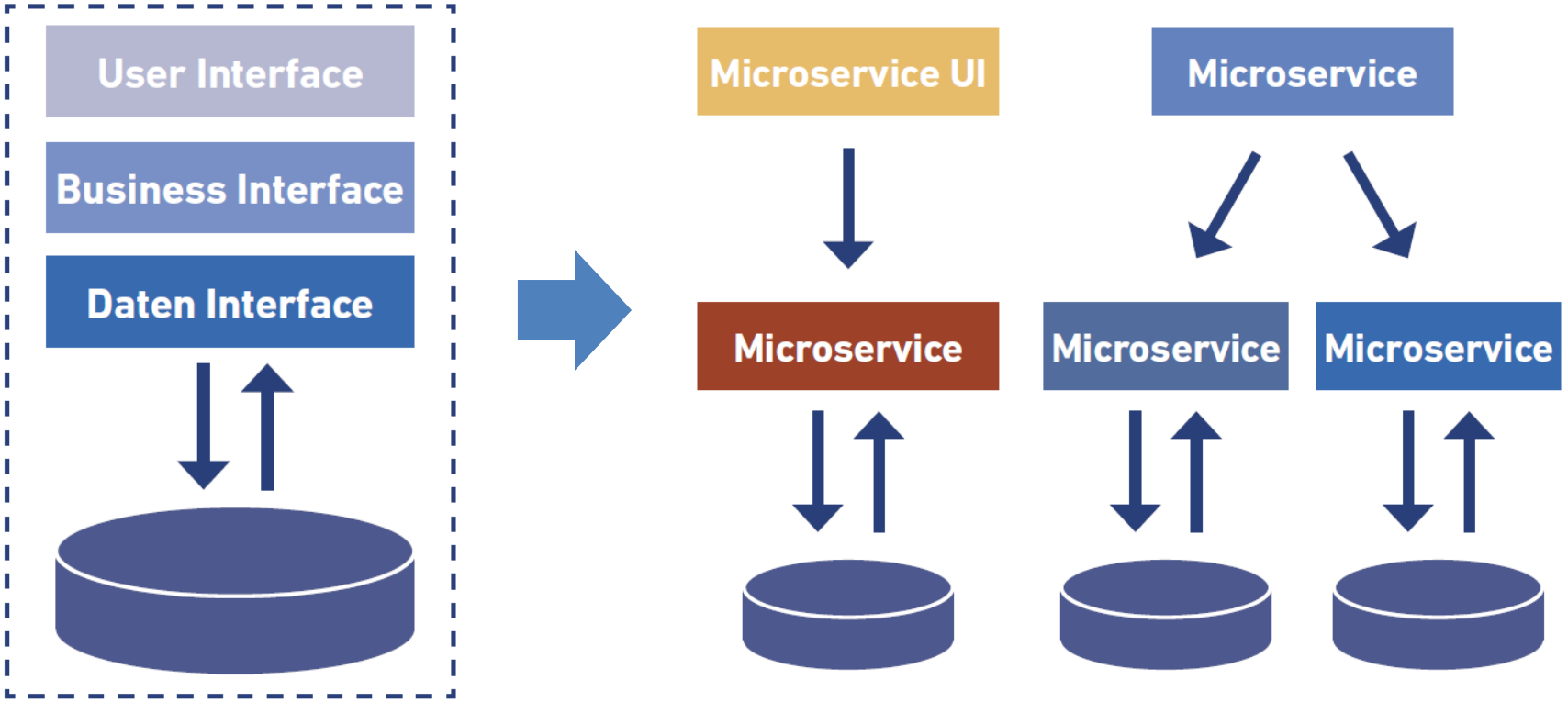


CloudMIG [Frey and Hasselbring 2010, 2011a, 2011b; Frey et al. 2013a, 2013b]

Migration to Microservices

Monolithic Architecture

Microservices Architecture



OBJEKTspektrum Microservices Poster, 2019

Agenda

1. Software Migration and Modernization
2. **Motivation for Migrating to Microservices**
3. Our Migration Case Study
4. Summary, so far

Drivers for Microservice Adoption

Table 5: Drivers for microservice adoption in different industries

	Overall		**Development / Consulting**	**Energy / Industry**	**Financial Services**	**Retail / E-Commerce**	**Other / Unknown**
	Score (mean)	SD	Score (mean)	Score (mean)	Score (mean)	Score (mean)	Score (mean)
High Scalability and Elasticity	2.14	0.72	2.19	2.18	2.15	2.67	1.89
High Maintainability	2.11	0.75	2.31	2.10	2.15	2.17	1.89
Short Time to Market	2.07	0.82	2.12	1.45	2.10	2.67	2.17
Enabler for CD and DevOps	1.61	0.89	1.69	1.27	1.70	1.83	1.56
Suitedness for Cloud and Docker	1.55	0.89	2.00	1.09	1.35	1.50	1.67
Organizational Improvement	1.37	0.81	1.31	1.18	1.50	1.83	1.22
Polyglot Programming	1.28	0.90	1.00	1.82	1.15	2.00	1.11
Polyglot Persistence	1.27	0.83	1.06	1.09	1.40	1.33	1.39
Attractiveness as Employer	0.87	0.86	1.00	0.55	0.85	1.67	0.72

Source: [Knoche & Hasselbring 2019]

See also: [Hasselbring 2016, 2018, Hasselbring & Steinacker 2017]

Modernization Goals

Table 9: Modernization goals for existing applications

	Overall		**Development / Consulting**	**Energy / Industry**	**Financial Services**	**Retail / E-Commerce**	**Other / Unknown**
	Score (mean)	SD	Score (mean)	Score (mean)	Score (mean)	Score (mean)	Score (mean)
Improve Maintainability	1.79	0.48	1.81	2.00	1.85	1.50	1.67
Improve Time to Market	1.54	0.63	1.88	1.09	1.60	1.83	1.33
Improve Scalability	1.46	0.58	1.44	1.64	1.60	1.50	1.22
Improve Quality	1.39	0.64	1.06	1.64	1.50	1.50	1.39
Prepare CD and DevOps	1.39	0.69	1.56	1.45	1.25	1.67	1.28
Introduce New Technology	1.04	0.75	0.88	1.09	0.90	1.50	1.17
Improve Team Motivation	0.97	0.68	0.94	0.82	1.10	1.50	0.78

Source: [Knoche & Hasselbring 2019]

Agenda

1. Software Migration and Modernization
2. Motivation for Migrating to Microservices
3. **Our Migration Case Study**
4. Summary, so far

Exemplar Legacy System

The exemplar legacy system…

- manages customer data of an insurance company
- was built in the 1970s and 1980s
- consists of ~1 million SLOC (COBOL)
- is part of a larger software ecosystem (COBOL and Java)

Why Modernize this System?

Primary drivers

- Lack of maintainability
- High time to market

Secondary drivers

- Vendor lock-in
- Programmer retirement

Architectural Modernization Goals

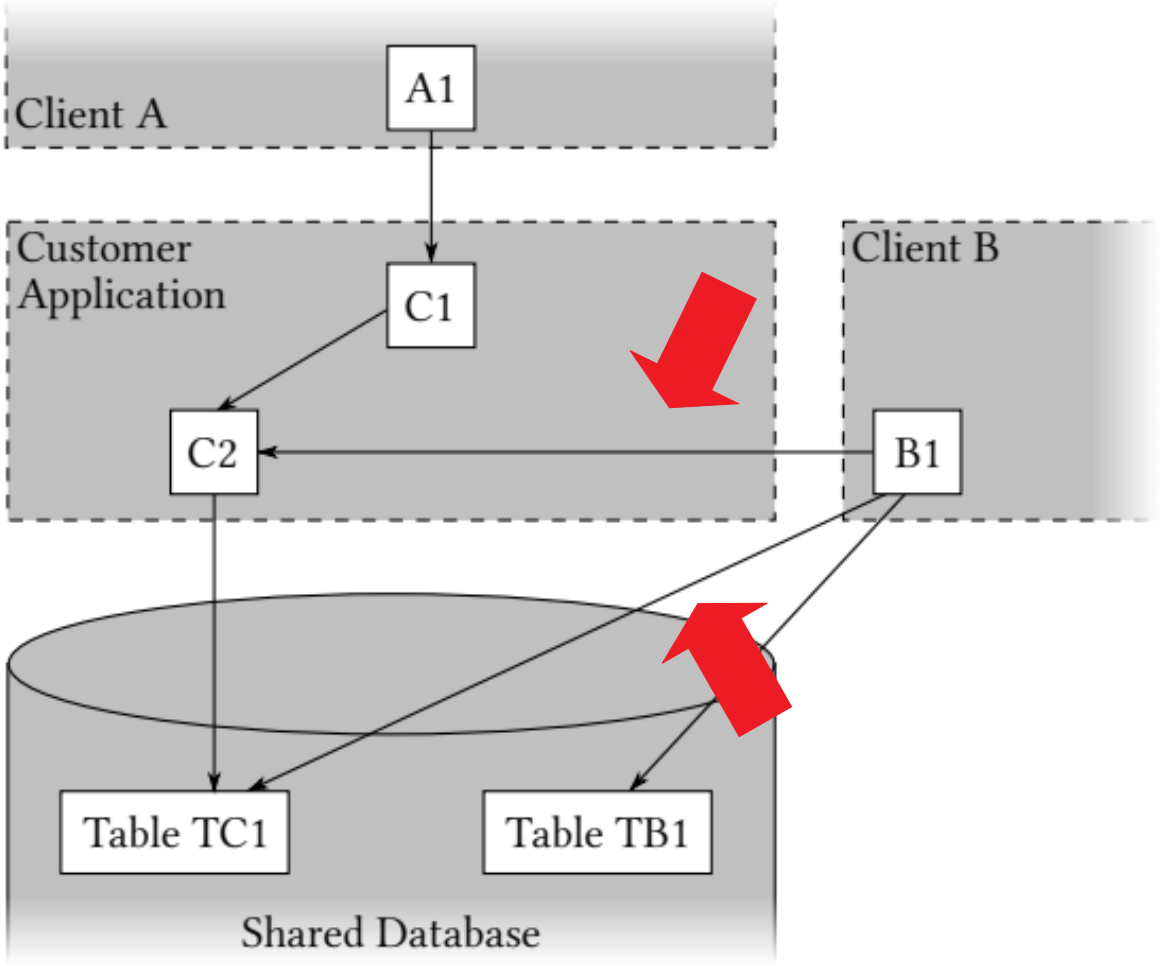
- Establish well-defined platform-independent interfaces
  - based on the **bounded contexts** of the underlying domain
- Reducing the number of entry points
- Eliminating redundant / obsolete parts of the application
- Incremental platform migration from COBOL to Java

Why Microservices to Achieve these Goals?

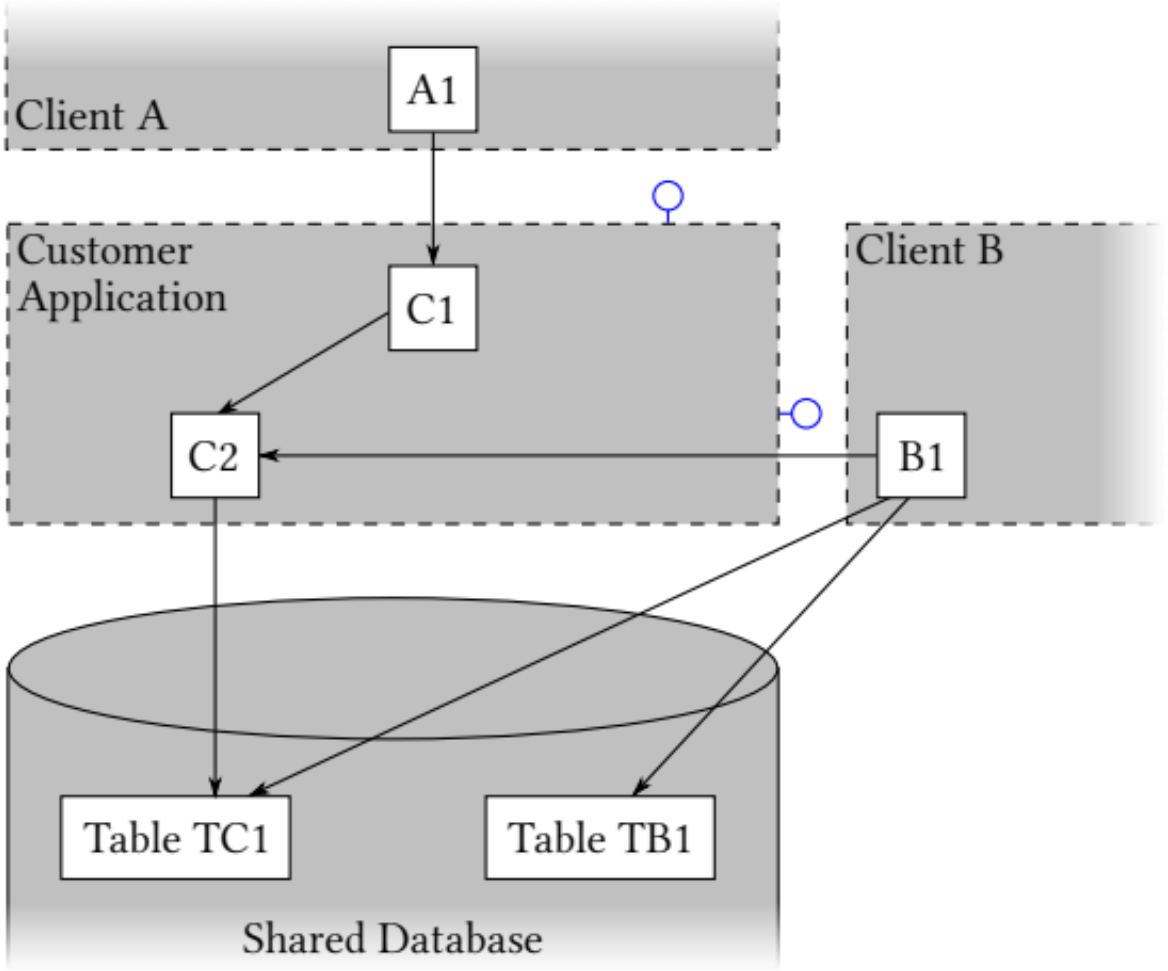
Microservices…

- provide strong component separation (including data)
- are independently deployable
- have independent lifecycles
- emphasize cross-platform interaction
- work well with CI / CD / DevOps

Migration Process: Step 0: Initial Situation



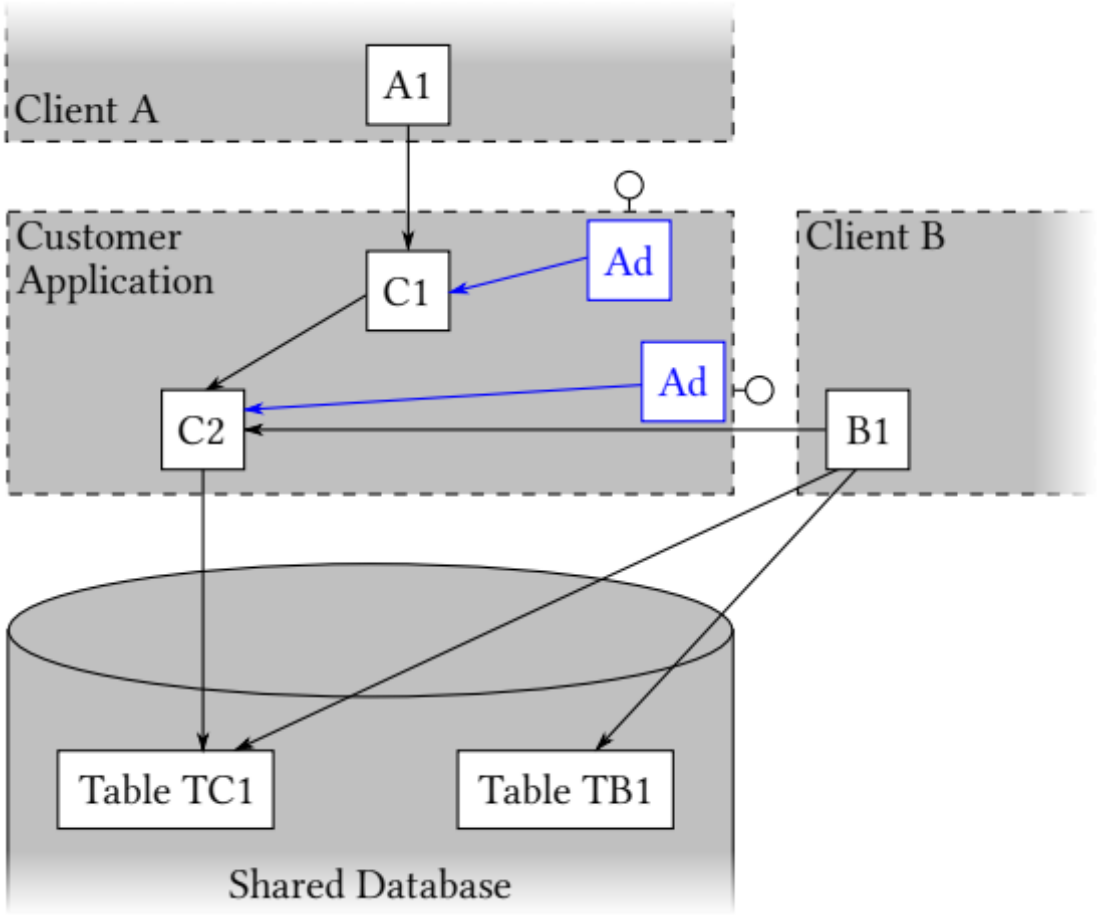
Migration Process: Step 1: Defining a Service Facade



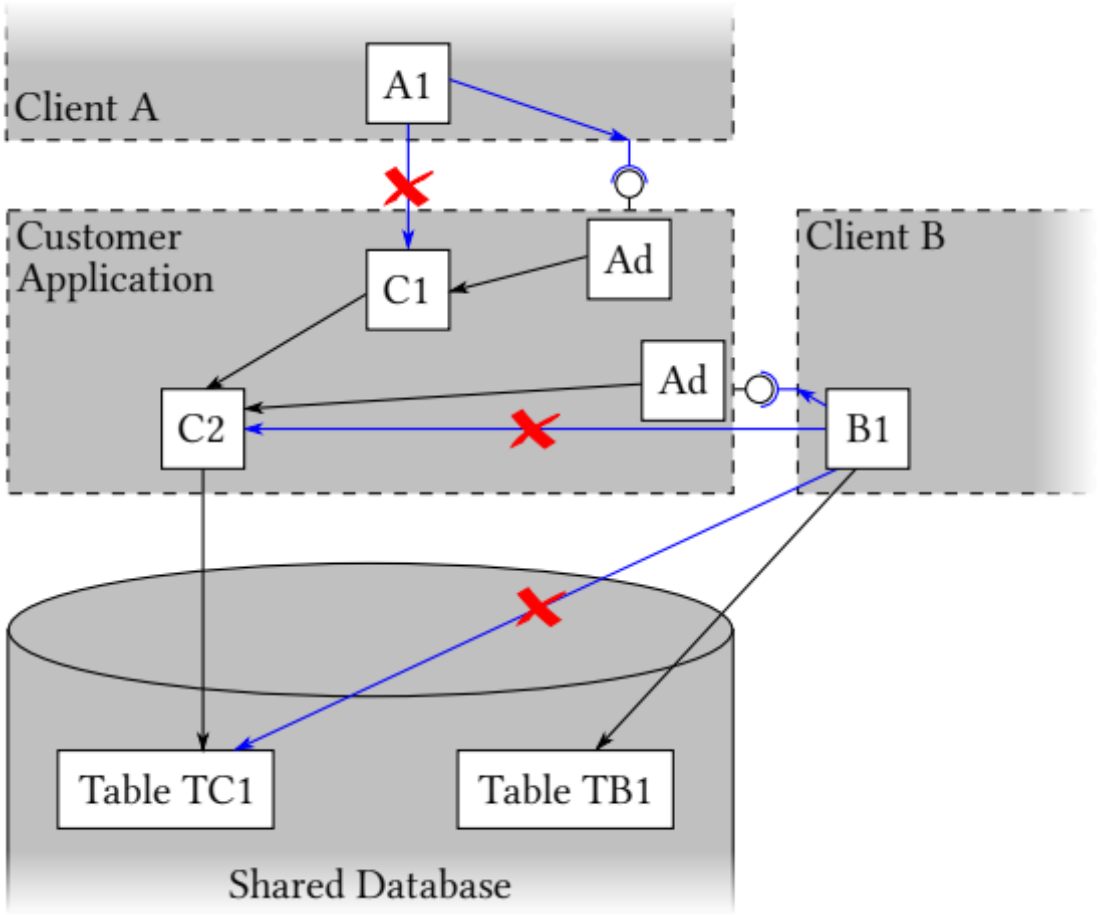
Defining a Service Facade

- First, a target domain model was designed
  - and used to define service operations from scratch.
- Afterward, static analysis was employed to identify the “entry points” of the existing application
  - i.e., programs, methods, or database tables that were accessed from other applications.
- Then, similar or redundant operations were merged

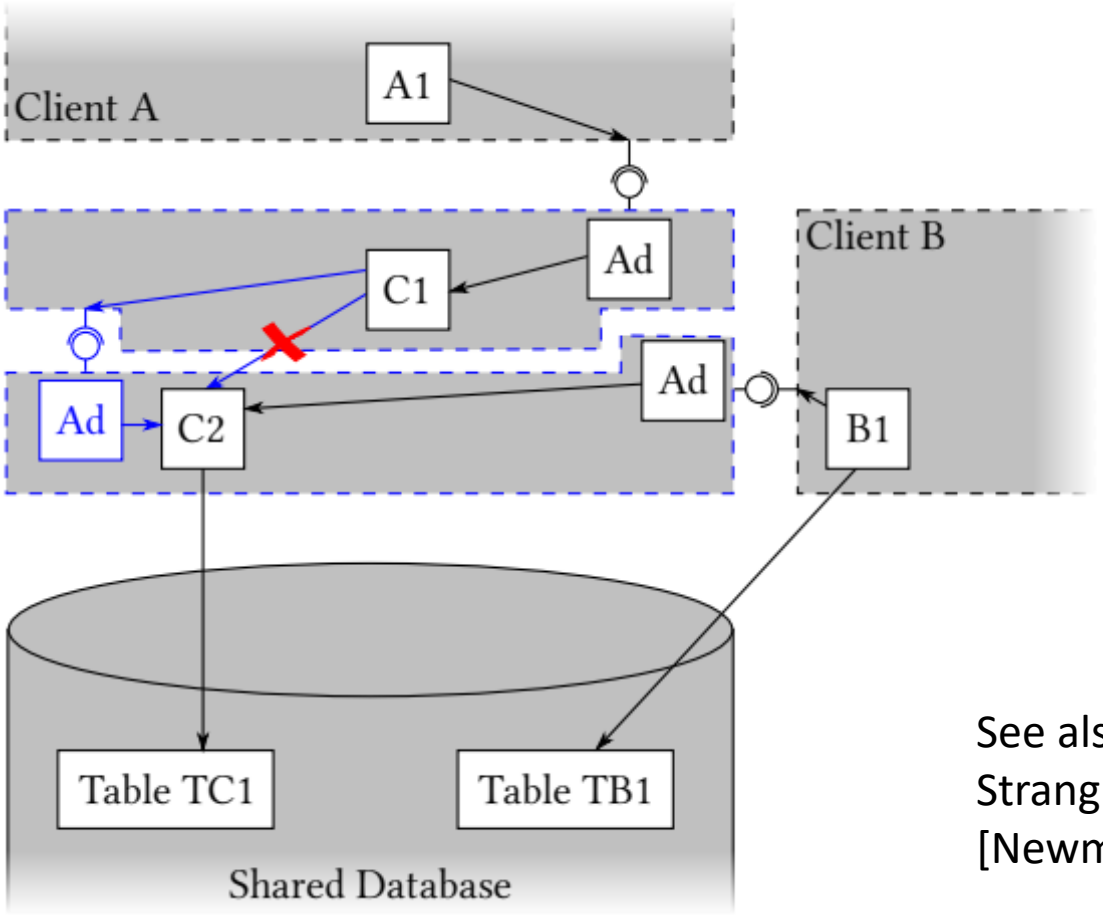
Migration Process: Step 2: Adapting the Service Facade



Migration Process: Step 3: Client Migration

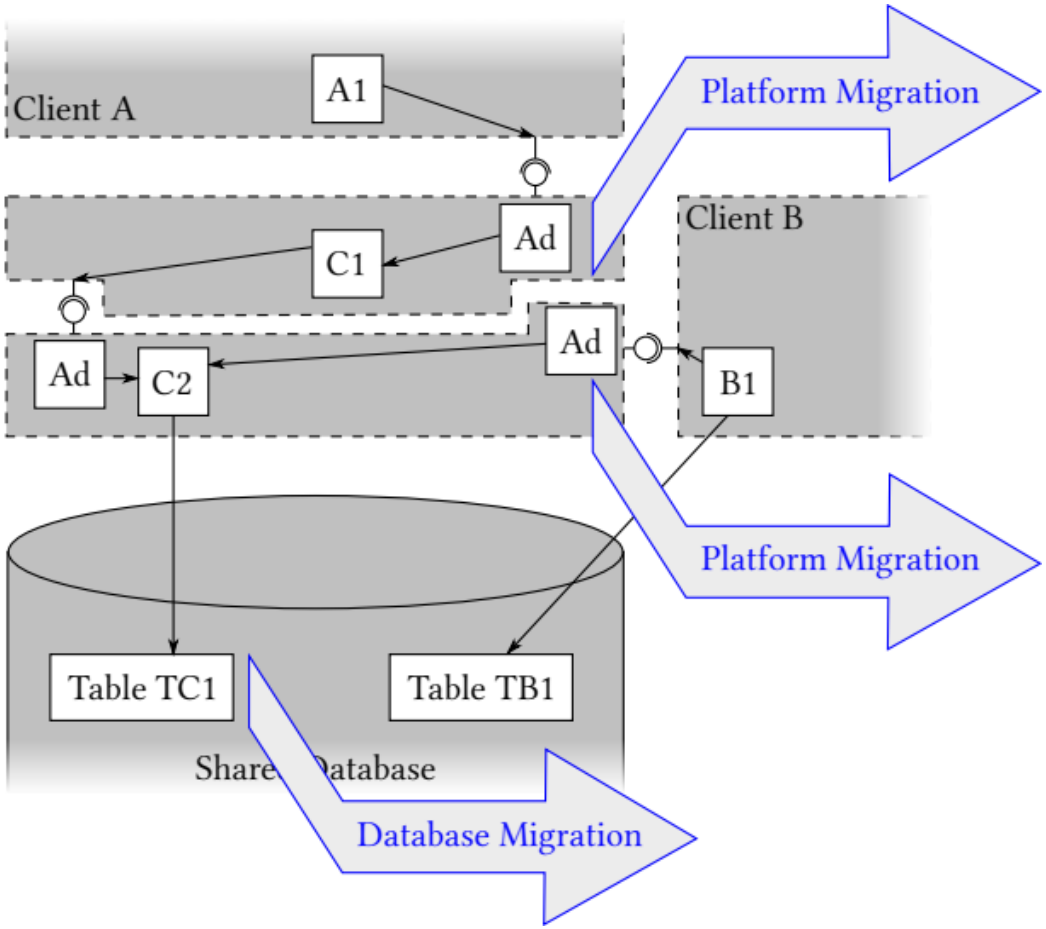


Migration Process: Step 4: Internal Restructuring



See also:
Strangler Pattern
[Newman 2015]

Migration Process: Step 5: Platform Migration



Current State

- What has been achieve so far?
  - The client migration has been decoupled from the backend migration.
    - Client migration is finished
  - First service operations have been migrated
  - New requirements were delivered on time
  - First legacy artifacts have been retired
  - Although the implementation is still based on the old COBOL code, it is now only accessed using well-defined, platform-independent interfaces.
  - In particular, the database has been decoupled such that, for instance, schema changes can now be performed without affecting client applications.
- Which challenges remain?
  - Database partitioning into (logical) bounded contexts
  - Transactions and Performance [Knoche 2016]
  - Batch jobs

Summary, for far

- A good modernization strategy delivers value even if some parts cannot be migrated
- Isolate risks, even if it means additional costs
- First careful steps toward infrastructure automation and **DevOps** practices,
  - as the new implementations create opportunities for experimenting with these approaches.
- It's not only technology
  - It's also about people
  - You always need a business case

References

[Frey and Hasselbring 2010] Frey, S. und Hasselbring, W., “Model-Based Migration of Legacy Software Systems to Scalable and Resource-Efficient Cloud-Based Applications: The CloudMIG Approach,” In: Proceedings of the First International Conference on Cloud Computing, GRIDs, and Virtualization (Cloud Computing 2010), 21-26 Nov 2010, Lisbon, Portugal.

[Frey and Hasselbring 2011a] Frey, S. and Hasselbring, W., “The CloudMIG Approach: Model-Based Migration of Software Systems to Cloud-Optimized Applications,” International Journal on Advances in Software, 4 (3 and 4). pp. 342-353.

[Frey and Hasselbring 2011b] Frey, S. and Hasselbring, W., “An Extensible Architecture for Detecting Violations of a Cloud Environment's Constraints During Legacy Software System Migration,” In: Proceedings of the 15th European Conference on Software Maintenance and Reengineering (CSMR 2011), 1-4 March 2011, Oldenburg, Germany.

[Frey et al. 2013a] Frey, S., Fittkau, F. und Hasselbring, W., “Search-Based Genetic Optimization for Deployment and Reconfiguration of Software in the Cloud,” In: 35th International Conference on Software Engineering (ICSE 2013), 18.-26. May 2013, San Francisco, CA, USA.

[Frey et al. 2013b] Frey, S., Hasselbring, W. und Schnoor, B., “Automatic Conformance Checking for Migrating Software Systems to Cloud Infrastructures and Platforms,” Journal of Software: Evolution and Process, 25 (10). pp. 1089-1115. DOI 10.1002/smr.582.

[Hasselbring 2016] W. Hasselbring, “Microservices for Scalability (Keynote Presentation),” In: 7th ACM/SPEC International Conference on Performance Engineering (ACM/SPEC ICPE 2016), March 15, 2016 , Delft, NL.

[Hasselbring 2018] W. Hasselbring, “Software Architecture: Past, Present, Future,” In: The Essence of Software Engineering. Springer, pp. 169-184. 2018. DOI 10.1007/978-3-319-73897-0_10

References

[Hasselbring et al. 2004] Hasselbring, W., Reussner, R. H., Jaekel, H., Schlegelmilch, J., Teschke, T. und Krieghoff, S., "The Dublo Architecture Pattern for Smooth Migration of Business Information Systems," In: Proceedings of the 26th International Conference on Software Engineering (ICSE 2004), 23-28 May 2004, Edinburgh, Scotland, UK.

[Hasselbring et al. 2008] Hasselbring, W., Büdenbender, A., Stefan, G., Krieghoff, S. and Marz, J., "Muster zur Migration betrieblicher Informationssysteme," In: Tagungsband Software Engineering 2008.

[Hasselbring & Steinacker 2017] W. Hasselbring, G. Steinacker: "Microservice Architectures for Scalability, Agility and Reliability in E-Commerce", In: Proceedings of the IEEE International Conference on Software Architecture (ICSA 2017), April 2017, Gothenburg, Sweden.

[Knoche 2016] H. Knoche: "Sustaining Runtime Performance while Incrementally Modernizing Transactional Monolithic Software towards Microservices," In: International Conference on Performance Engineering (ICPE 2016), March 14, 2016, Delft, Netherlands. 2016

[Knoche and Hasselbring 2018] H. Knoche and W. Hasselbring, "Using Microservices for Legacy Software Modernization," IEEE Software, 35 (3). pp. 44-49. 2018. DOI 10.1109/MS.2018.2141035.

[Knoche and Hasselbring 2019] H. Knoche and W. Hasselbring, "Drivers and Barriers for Microservice Adoption - A Survey among Professionals in Germany," Enterprise Modelling and Information Systems Architectures (EMISAJ) - International Journal of Conceptual Modeling, 14 (1). pp. 1-35. 2019. DOI https://doi.org/10.18417/emisa.14.1.

[Newman 2015] Newman, S., "Building microservices: designing fine-grained systems," O'Reilly. 2015

[van Hoorn et al. 2011] van Hoorn, A., Frey, S., Goerigk, W., Hasselbring, W., Knoche, H., Köster, S., Krause, H., Porembski, M., Stahl, T., Steinkamp, M. and Wittmüss, N. "DynaMod Project: Dynamic Analysis for Model-Driven Software Modernization," In: 1st International Workshop on Model-Driven Software Migration (MDSM), 2011