Vorhersagbare Migration in Echtzeitsystemen

Fachgruppentreffen Betriebssysteme, 11.03.2021

Phillip Raffeck[1], Stefan Reif[1], Wolfgang Schröder-Preikschat[1], Peter Ulbrich[2]

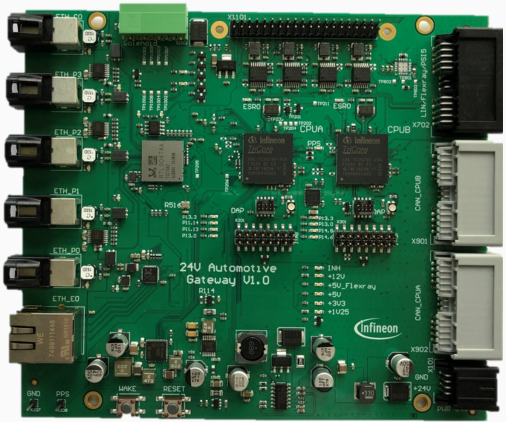
[1]Friedrich-Alexander-Universität Erlangen-Nürnberg
[2]Technische Universität Dortmund

technische universität dortmund

DFG

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

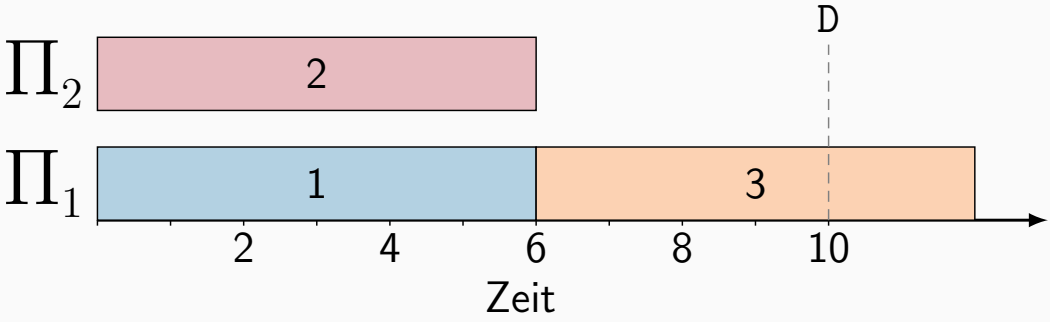
Komplexe, eingebettete Systeme



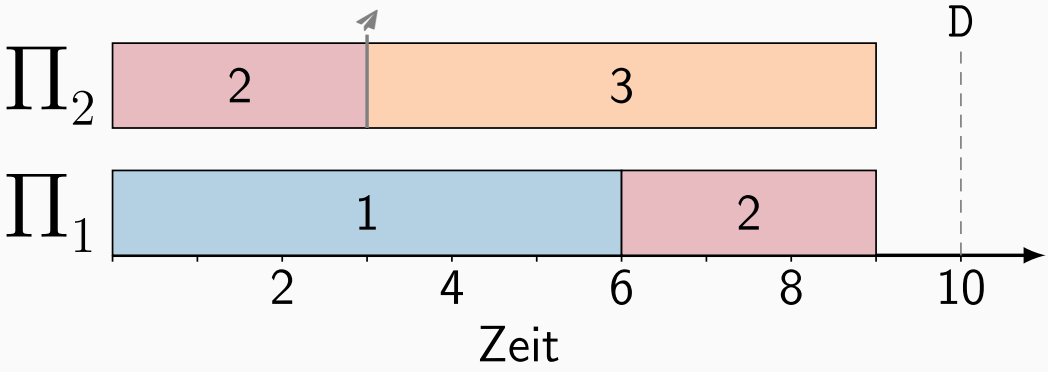
Quelle: Infineon

https://www.infineon.com/cms/en/product/evaluation-boards/kit_a2g_tc397_24v_gtw

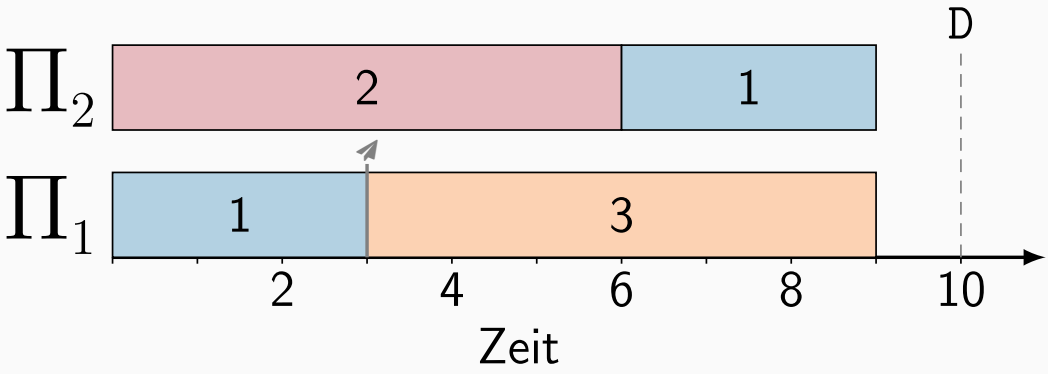
Migration als Notwendigkeit



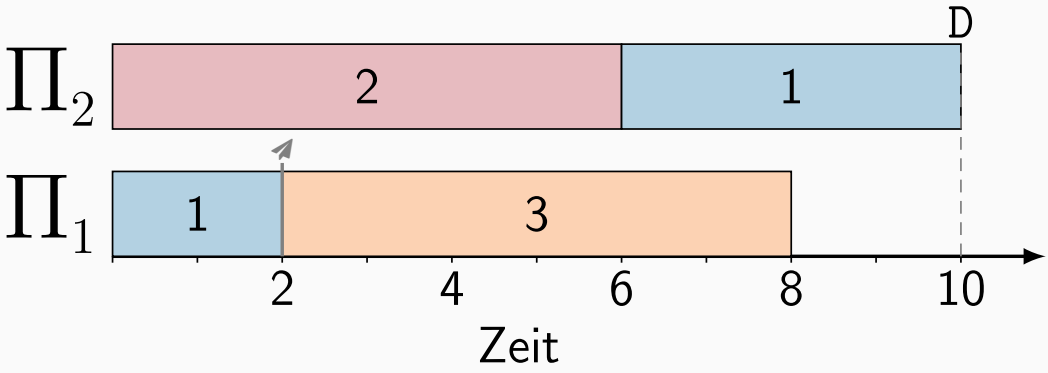
Unabwägbarkeit



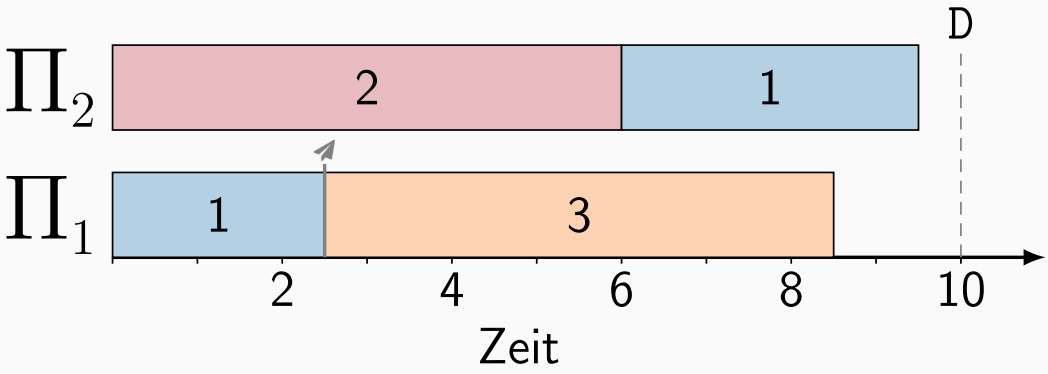
Unabwägbarkeit



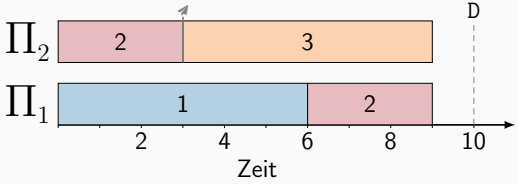
Unabwägbarkeit

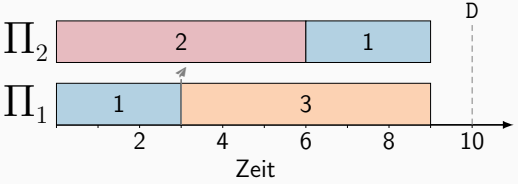


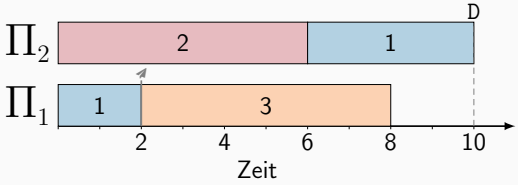
Unabwägbarkeit

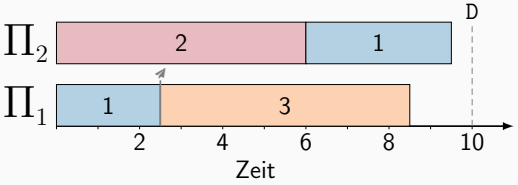


Unabwägbarkeit

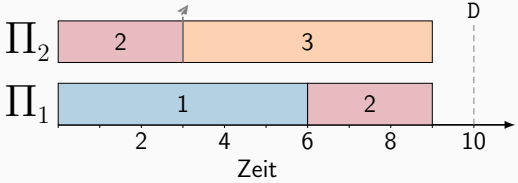


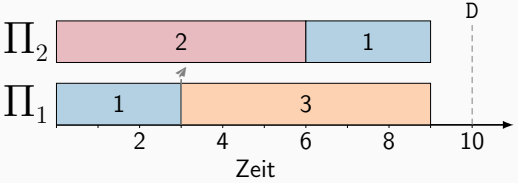




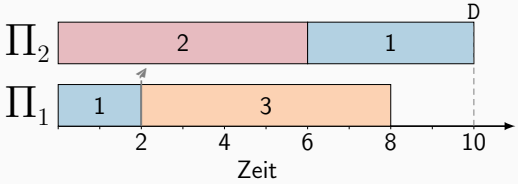


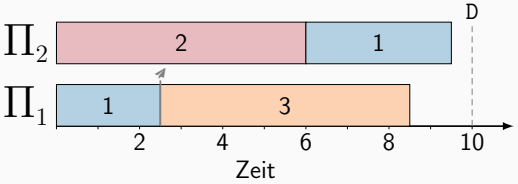
Unabwägbarkeit





Migration = Migration?





Speicherzugriffslatenzen

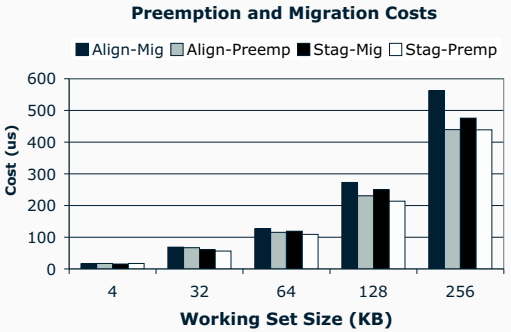
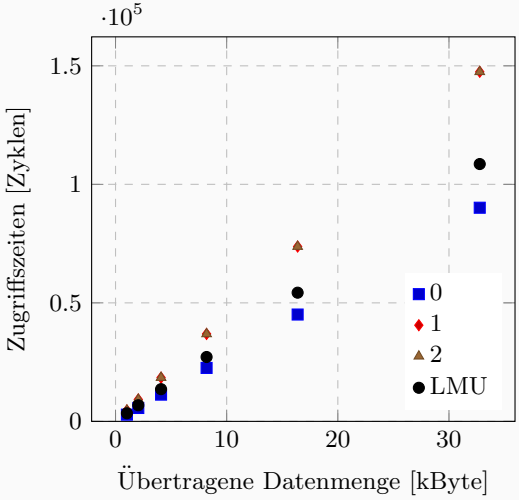
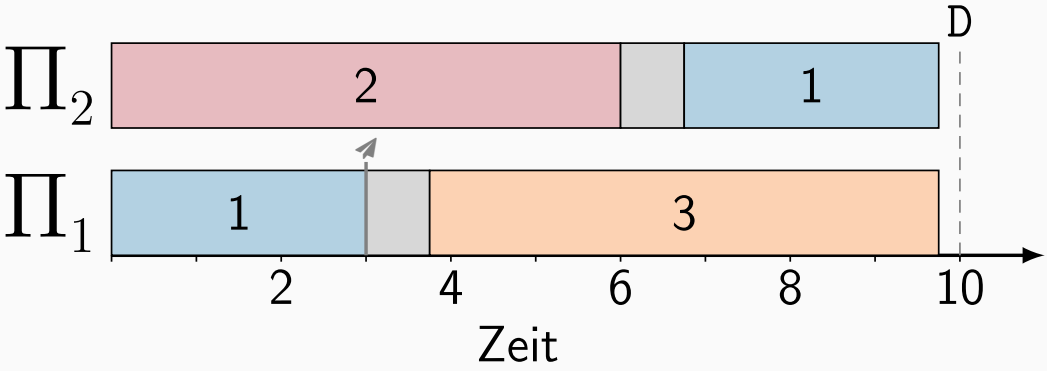


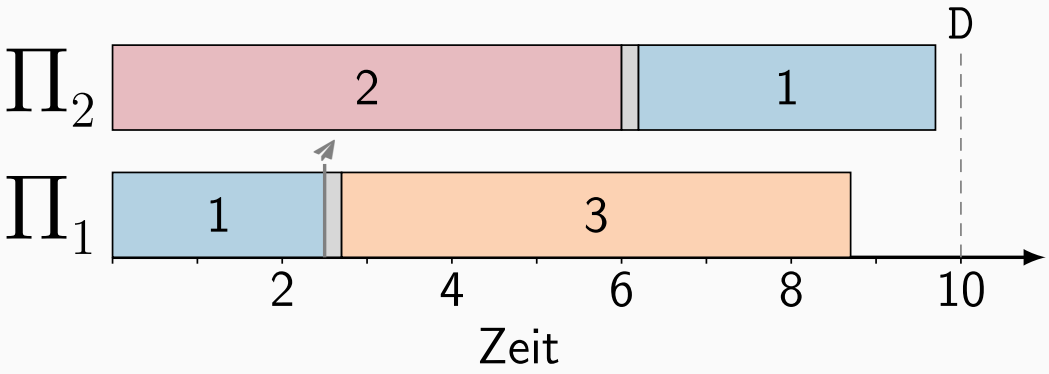
Abbildung aus: Calandrino, J, et al. LITMUSRT: A Testbed for Empirically Comparing Real-Time Multiprocessor Schedulers, RTSS'06



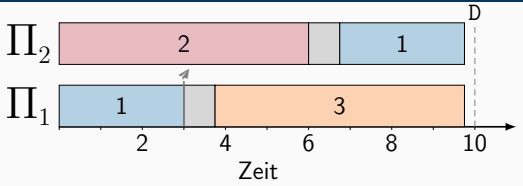
Verwaltungsgemeinkosten

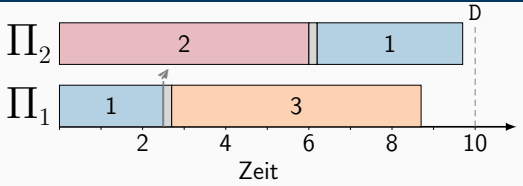


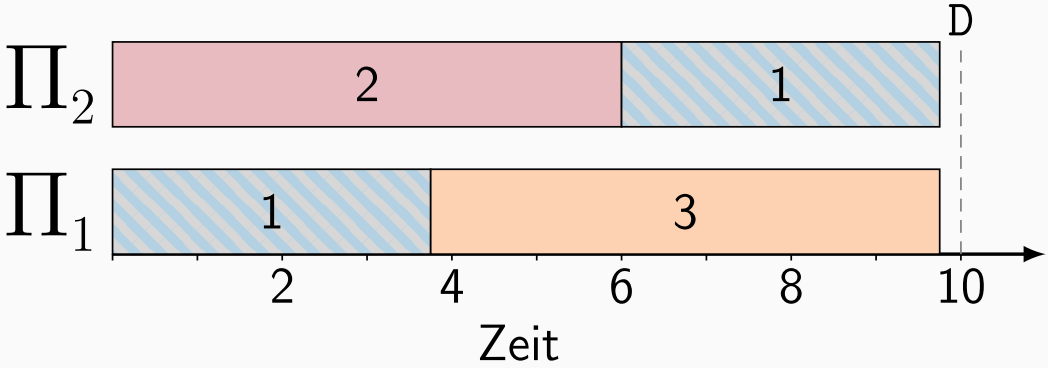
Verwaltungsgemeinkosten



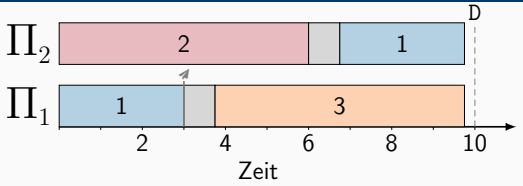
Verwaltungsgemeinkosten

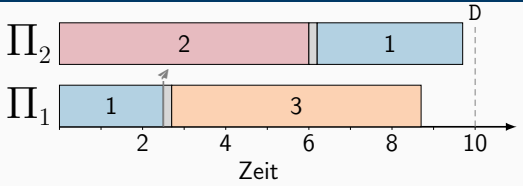


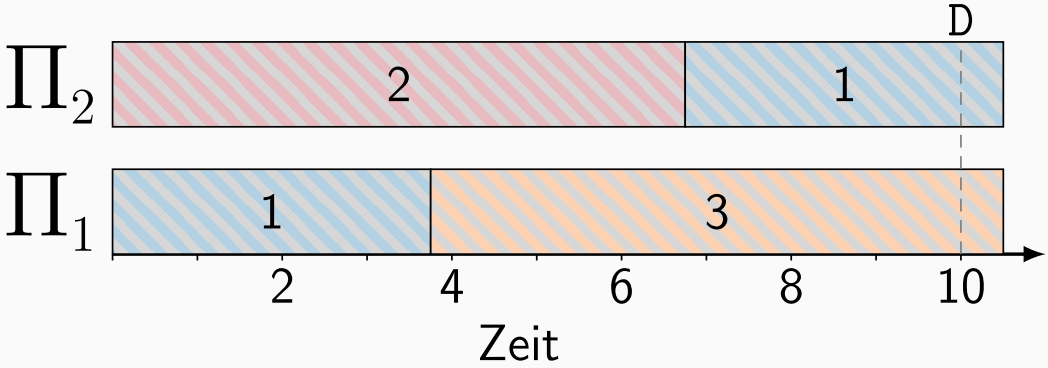




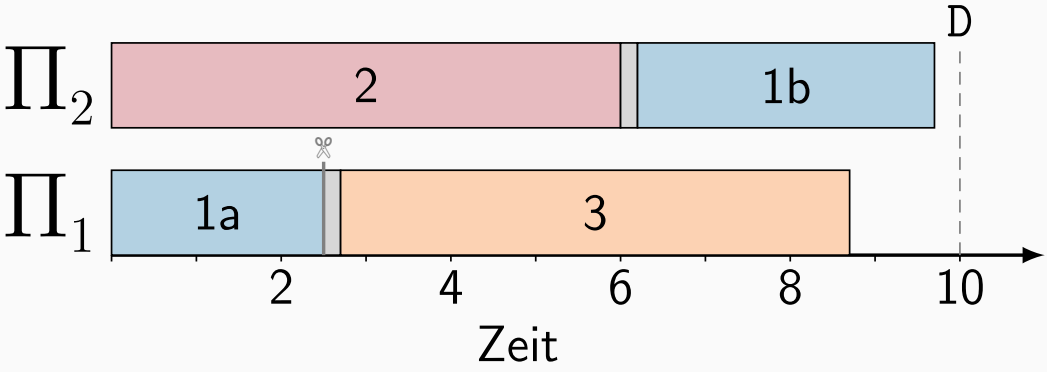
Verwaltungsgemeinkosten



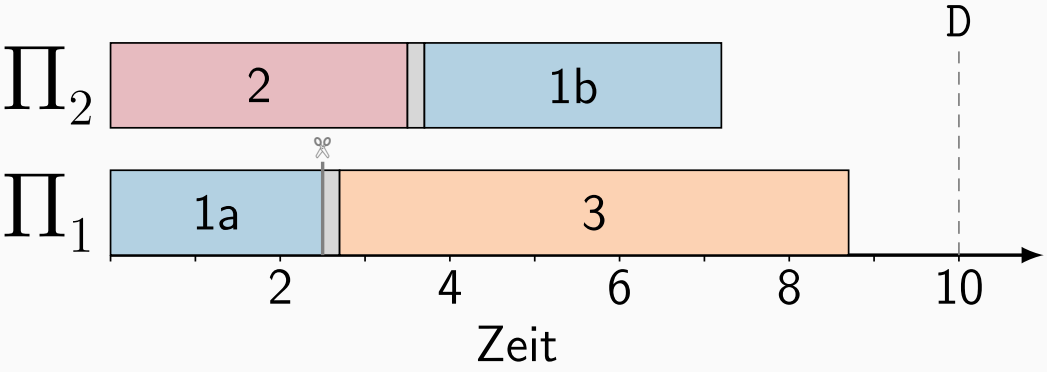




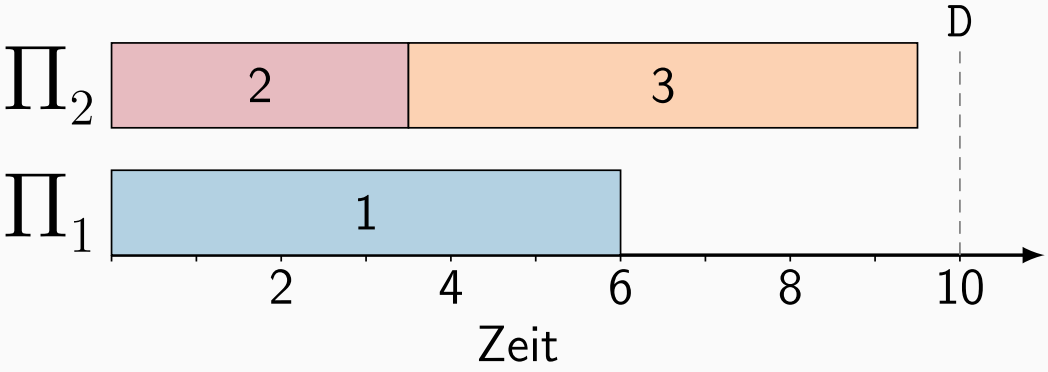
Statisch gebilligte Migration



Statisch gebilligte Migration



Statisch gebilligte Migration



Ziele

- Vorhersagbarkeit: Finden günstiger Migrationspunkte
- Flexibilität: Vermeidung unnötiger Migration
- Laufzeiteffizienz

Agenda

Migration in Echtzeitsystemen

Vorhersagbarkeit

Betriebssystemmechanismen

Migrationsbasierte Synchronisation

Fazit

Statische Festlegung

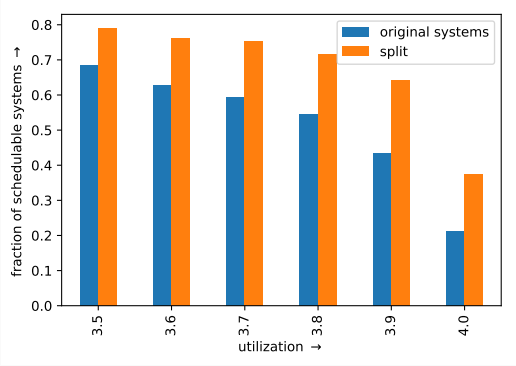
Bekannte Ansätze

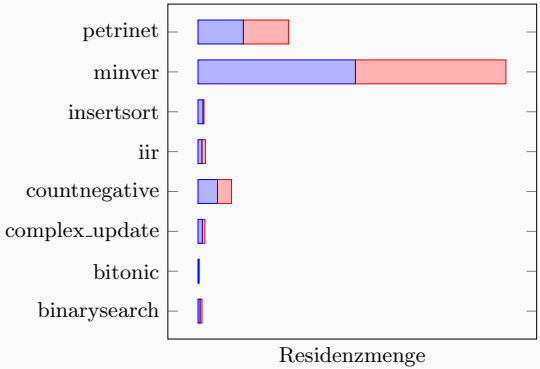
- automatisiertes Checkpointing
- implizit kooperatives Scheduling
- Platzierung von Migrationspunkten

Unser Ansatz

- multikriteriell: WCET und Migrationskosten
- gezielte Suche vs. gleichmäßiges Streuen
- Maßschneiderung

Erste Ergebnisse

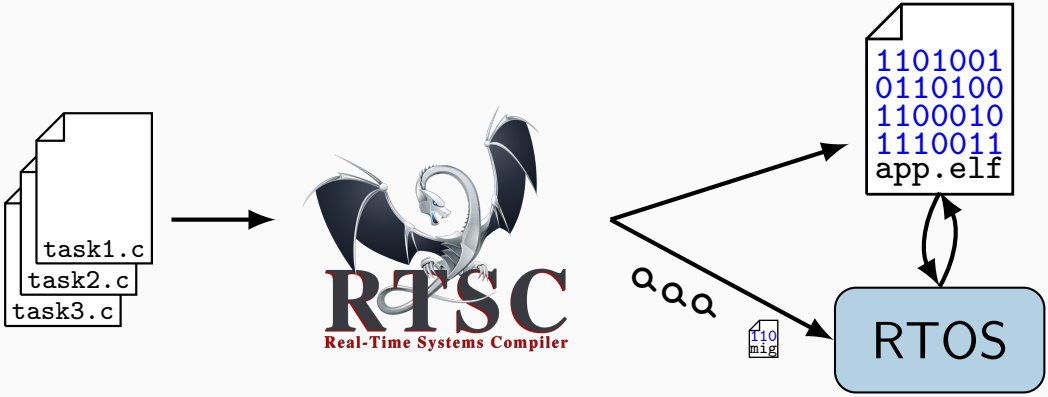




Klaus, T; Ulbrich, P; Raffeck, P; Frank, B; Wernet, L; Ritter von Onciul, M; Schröder-Preikschat, W. Boosting Job-Level Migration by Static Analysis, OSPERT'19

Betriebssystemmechanismen

Betriebssystemschnittstelle



Migrationsmechanismen

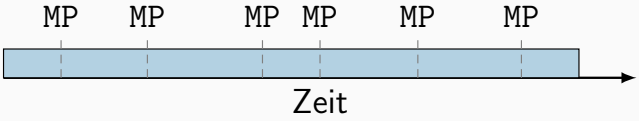
Vorbereitung

- Auslösen durch Anwendung
  - Systemaufruf
  - maßgeschneiderte Funktion
- eingewoben durch den Compiler
- sichere Kooperation

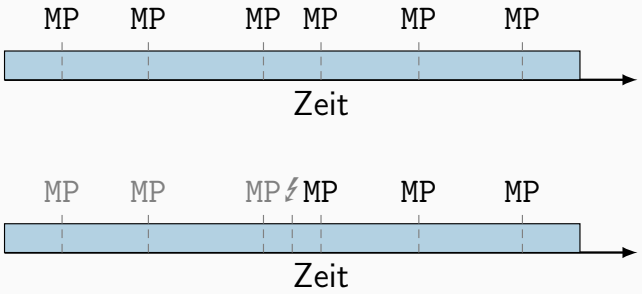
Zur Laufzeit

- Entscheidung nach Schlupf
- Überprüfung an Migrationspunkt
- Überprüfung nach Zeitspanne
- Überspringen unnötiger Punkte

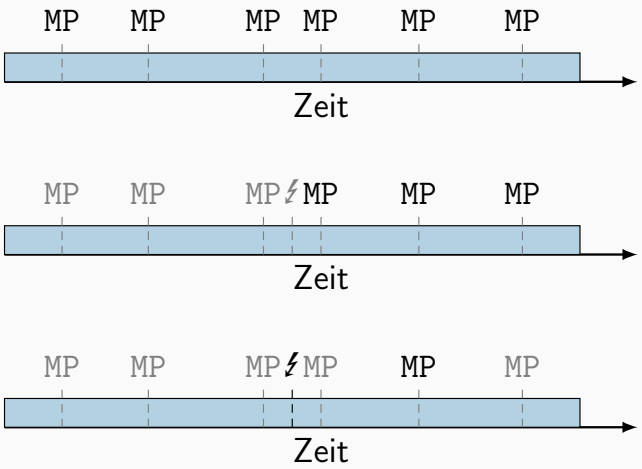
Mögliche Ablaufplanung zur Laufzeit



Mögliche Ablaufplanung zur Laufzeit



Mögliche Ablaufplanung zur Laufzeit



Wohin mit dem gesammelten Wissen?

1. Migration statt Verzögerung wartender Tasks
2. Erweiterung des Entscheidungsspielraums um Kritikalität
3. Migration als primäre Koordinationsmaßnahme

Migrationsbasierte Synchronisation

Synchronisation Globaler Datenstrukturen

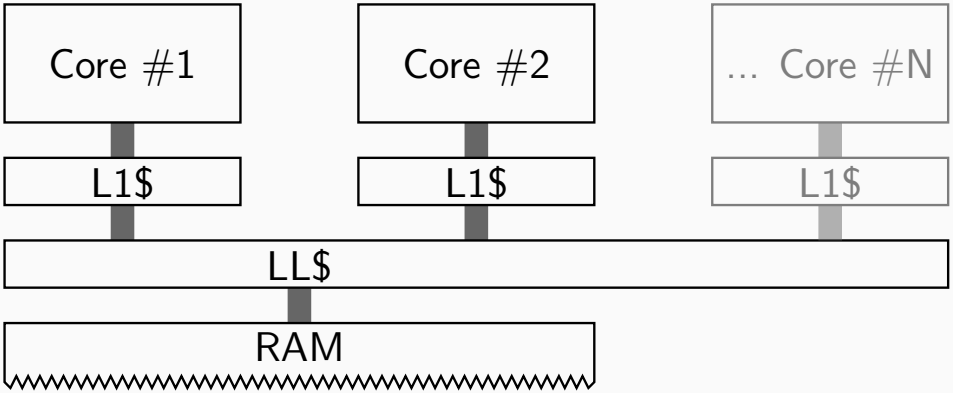
Lock-basierte Synchronisation

- schlechte Datenlokalität, häufige Cache-misses
- Blockadezeiten abhängig von WCET kritischer Abschnitte

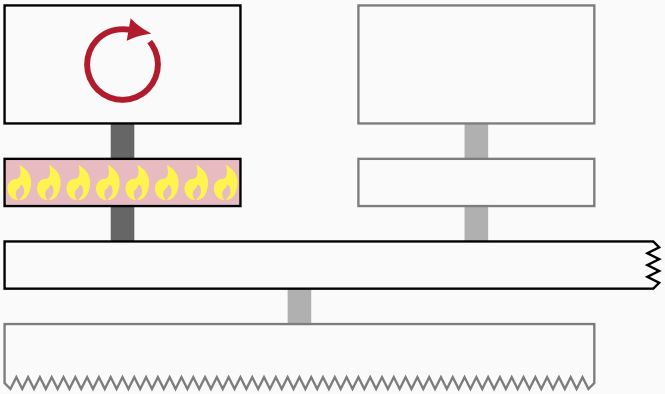
Migrationsbasierte Synchronisation

- Kontrollfluss- statt Datenmigration
- dedizierte „Synchronisationskerne”
- geteilte Daten in garantiert warmen Caches
- Schnittstellenkompatibel zu Locks

Migrationsbasierte Synchronisation

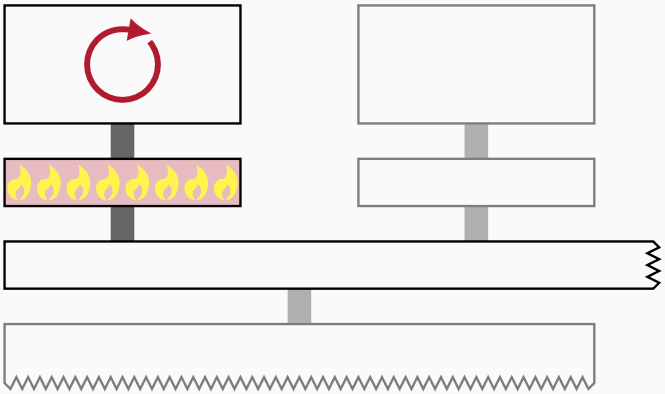


Migrationsbasierte Synchronisation



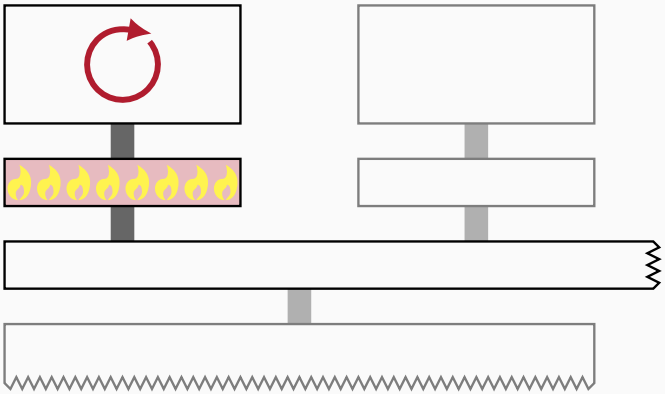
```
foo();
```

Migrationsbasierte Synchronisation



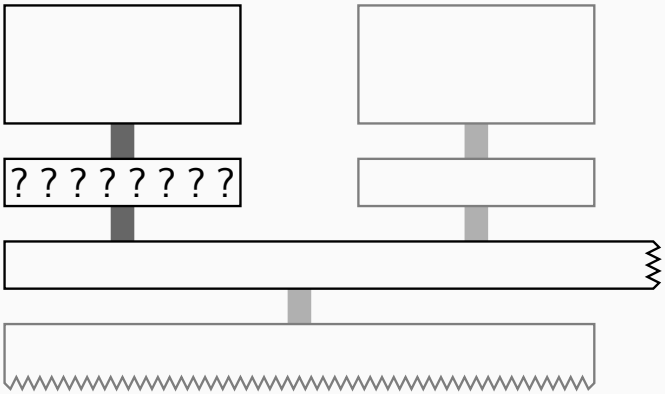
```
foo();
lock();

use(data);
```

Migrationsbasierte Synchronisation



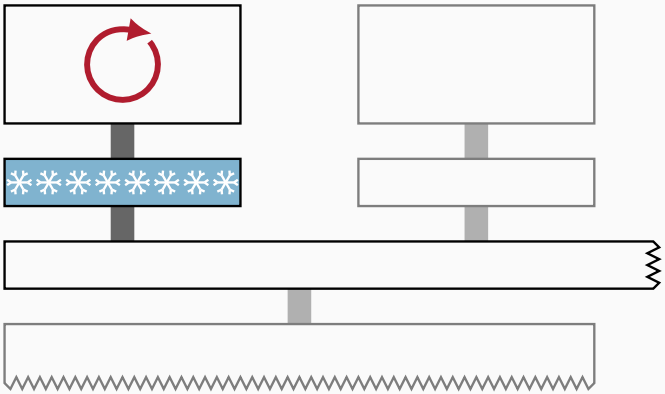
```
foo();
lock();
  ↳sleep();
use(data);
```

Migrationsbasierte Synchronisation



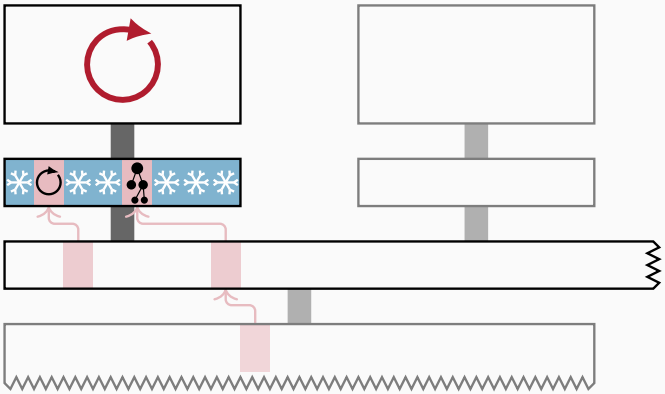

```
foo();
lock();
  ↳ sleep();
use(data);
```

Migrationsbasierte Synchronisation



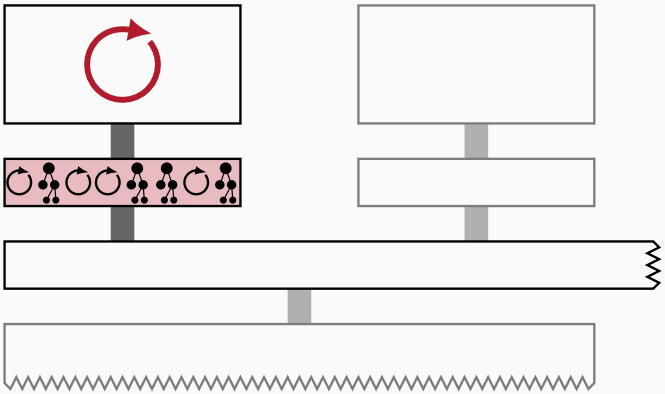
```
foo();
lock();
  └▸sleep();
▸use(data);
```

Migrationsbasierte Synchronisation



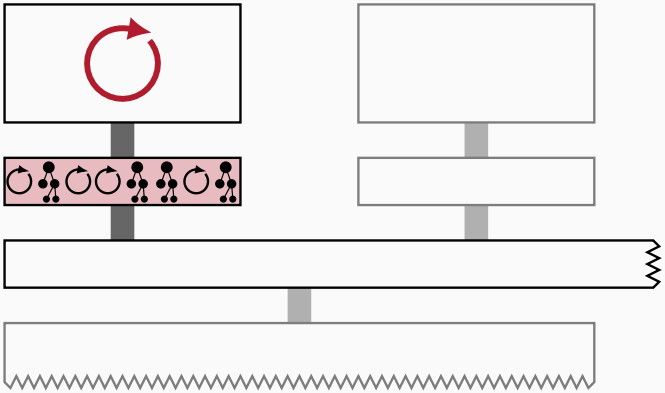
```
foo();
lock();
  └▸sleep();
use(data);
```

Migrationsbasierte Synchronisation



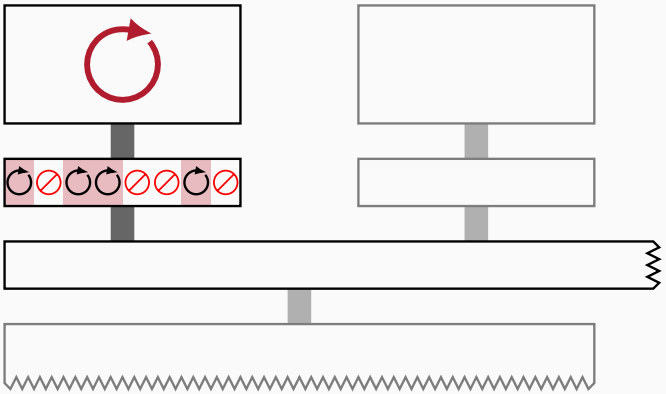
```
foo();
lock();
  └▸sleep();
use(data);
```

Migrationsbasierte Synchronisation



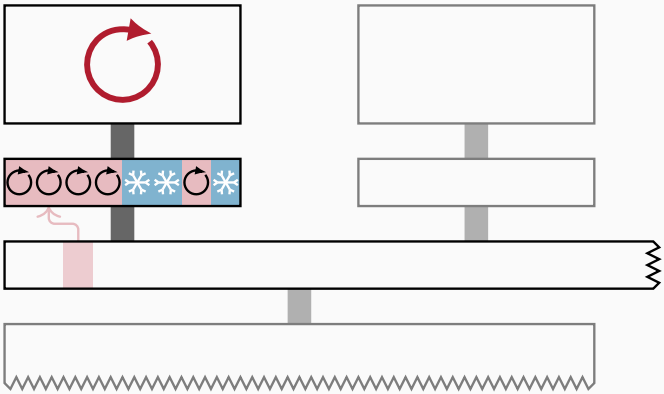
```
foo();
lock();
  └▸sleep();
use(data);
unlock();
```

Migrationsbasierte Synchronisation



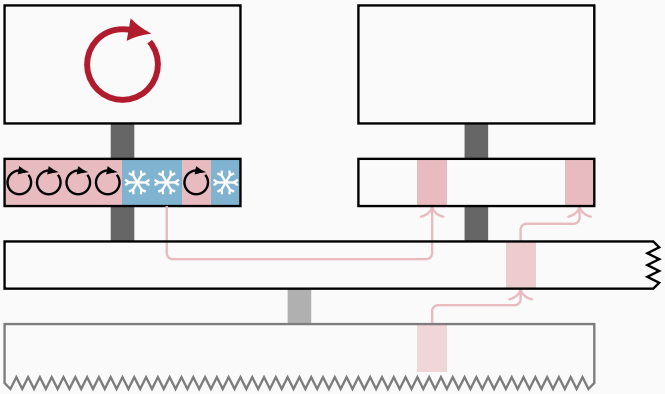
```
foo();
lock();
  └→sleep();
use(data);
→unlock();
```

Migrationsbasierte Synchronisation



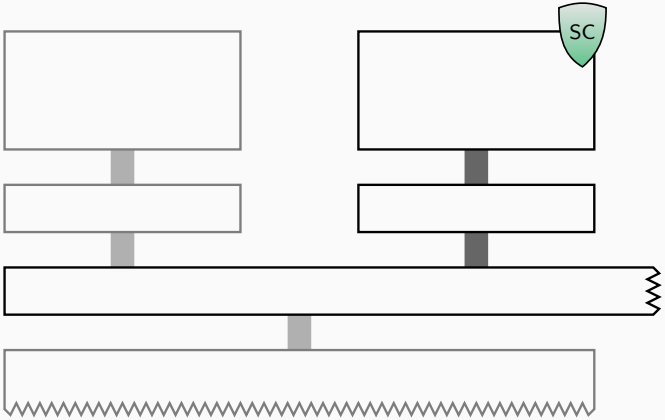
```
foo();
lock();
  └▸sleep();
use(data);
unlock();
bar();
```

Migrationsbasierte Synchronisation

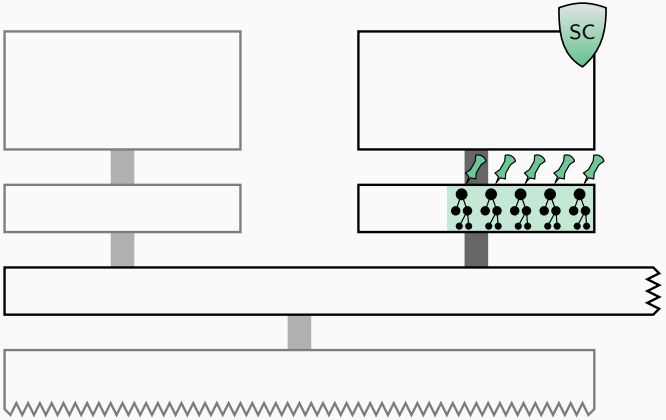


```
foo();
lock();
└▸sleep();
use(data);
unlock();
bar();
```

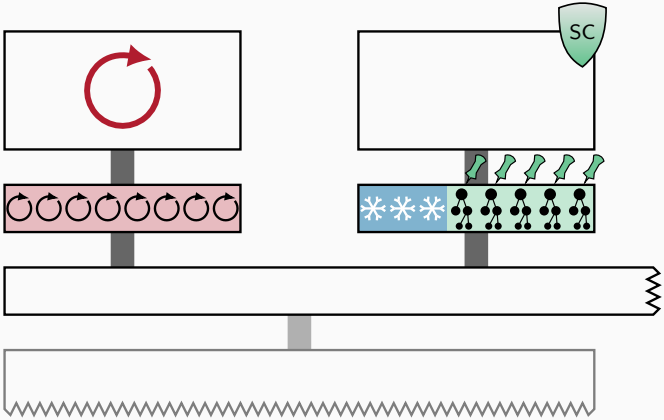
Migrationsbasierte Synchronisation



Migrationsbasierte Synchronisation

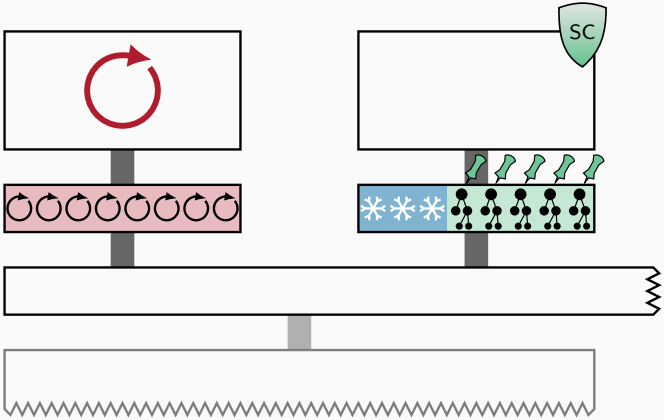


Migrationsbasierte Synchronisation



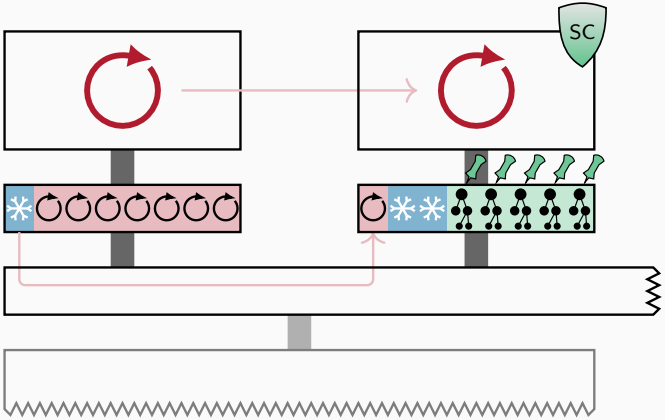
foo();

Migrationsbasierte Synchronisation



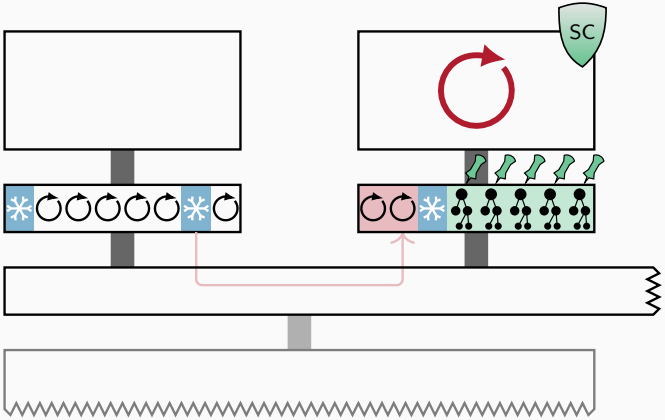

```
foo();
lock();

use(data);
```

Migrationsbasierte Synchronisation



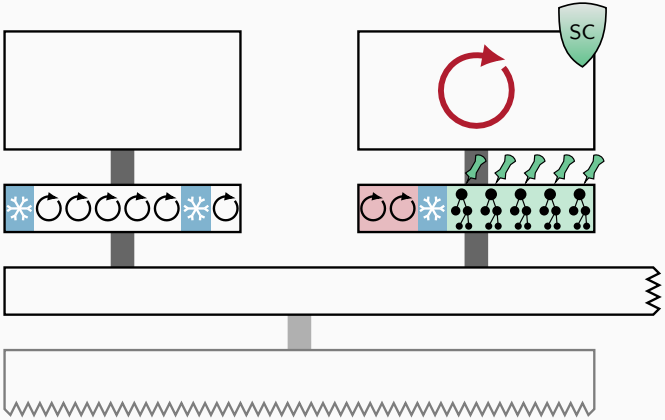

```
foo();
lock();
  └migrate(sc);
use(data);
```

Migrationsbasierte Synchronisation



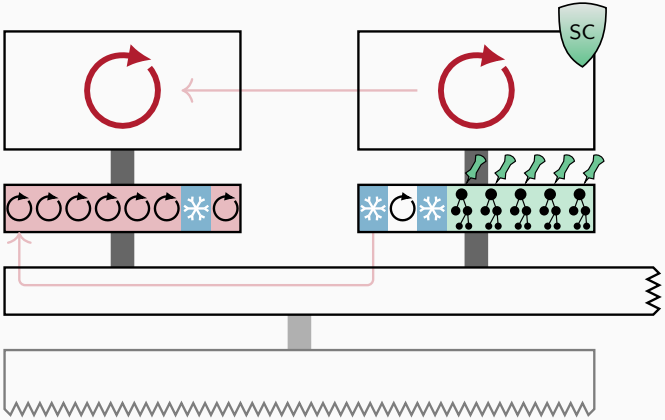

```
foo();
lock();
└▸migrate(sc);
use(data);
```

Migrationsbasierte Synchronisation



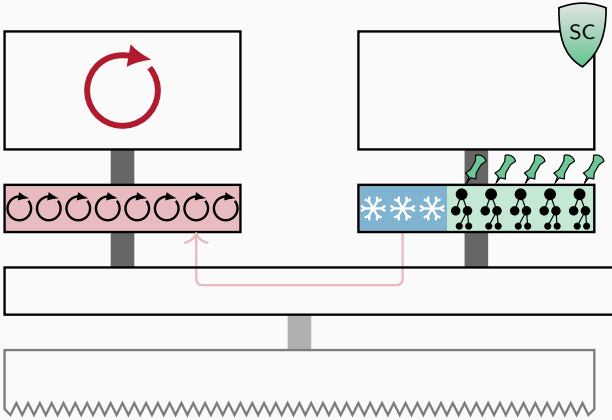

```
foo();
lock();
└►migrate(sc);
use(data);
unlock();
```

Migrationsbasierte Synchronisation



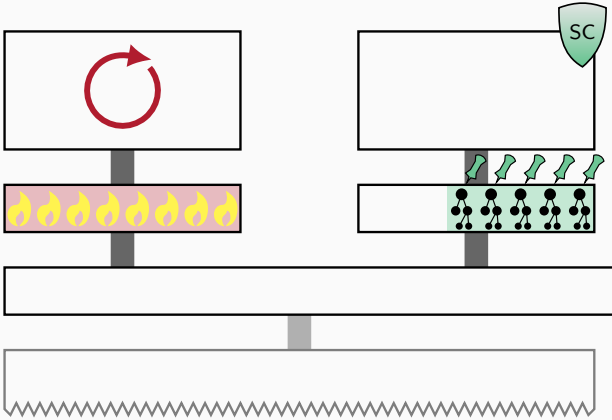

```
foo();
lock();
  └migrate(sc);
use(data);
unlock();
  └migrate(back);
```

Migrationsbasierte Synchronisation




```
foo();
lock();
└migrate(sc);
use(data);
unlock();
└migrate(back);
bar();
```

Migrationsbasierte Synchronisation




```
foo();
lock();
└migrate(sc);
use(data);
unlock();
└migrate(back);
bar();
```

Fazit

Fazit

- vorhersagbare Migration
- Laufzeitentscheidungen basierend auf Hinweisen
- Werkzeugunterstützung nötig
- geschickte Ausnutzung der Möglichkeiten
- effizientere, gut vorhersagbare Ausführung