# Mikrotjänster och distribuerad mjukvara

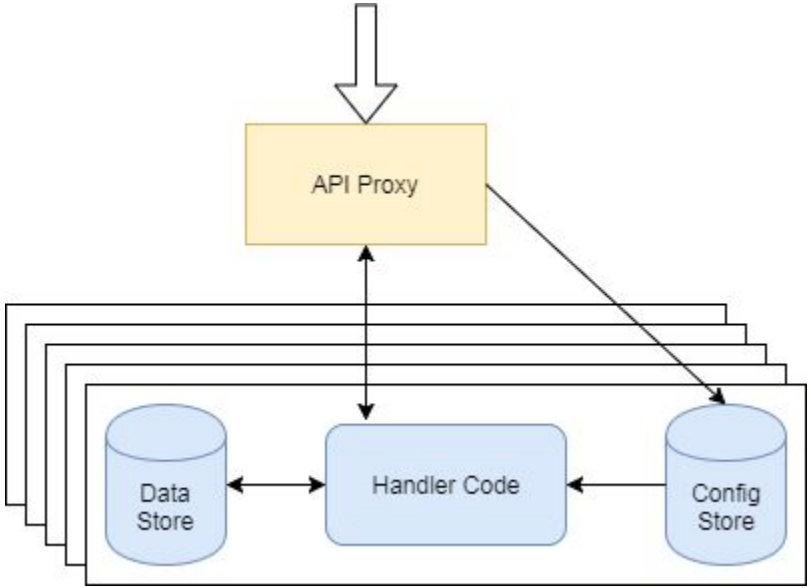
Bryt monoliten!

Richard Tyregrim

Senior Solutions Architect/SME, Software Strategist

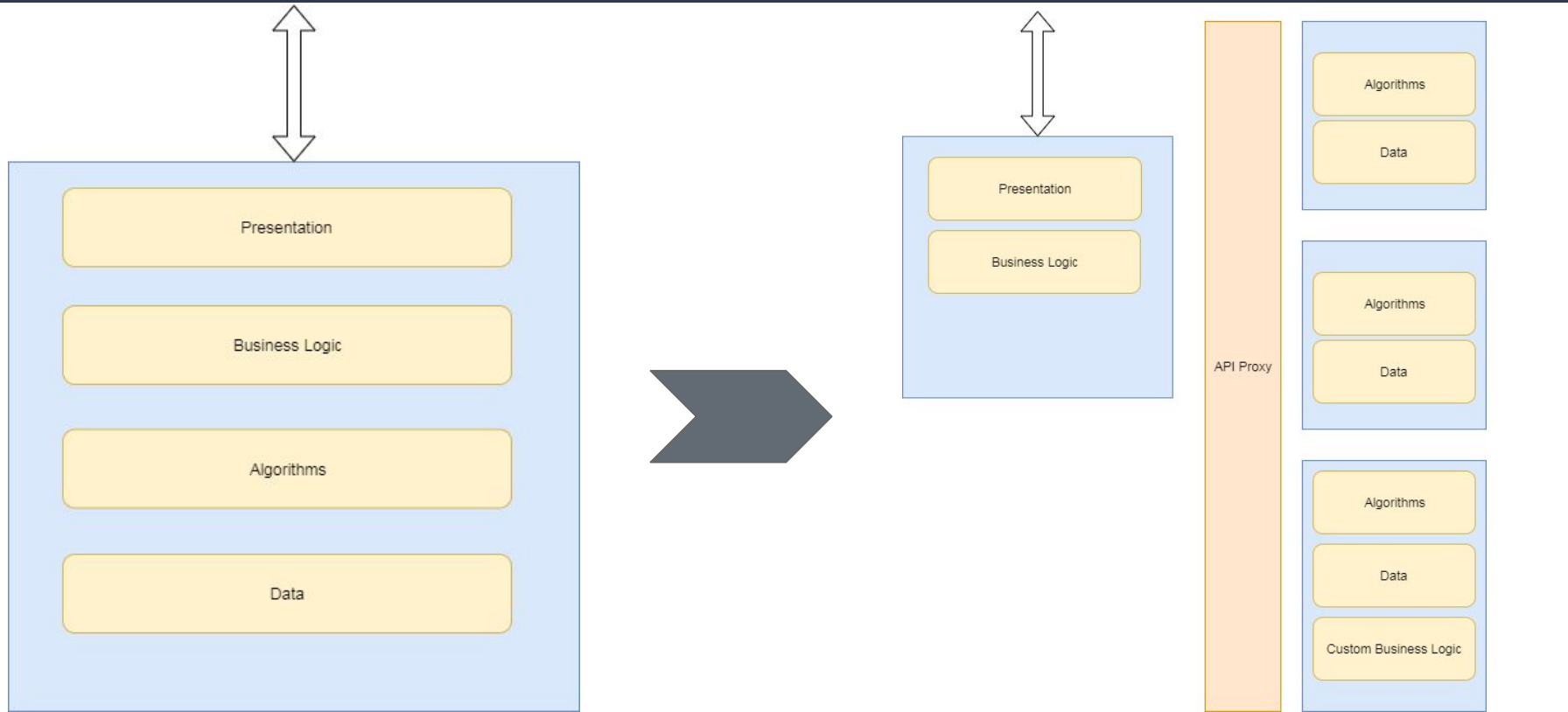

https://www.linkedin.com/in/richard-tyregrim-b4319734/

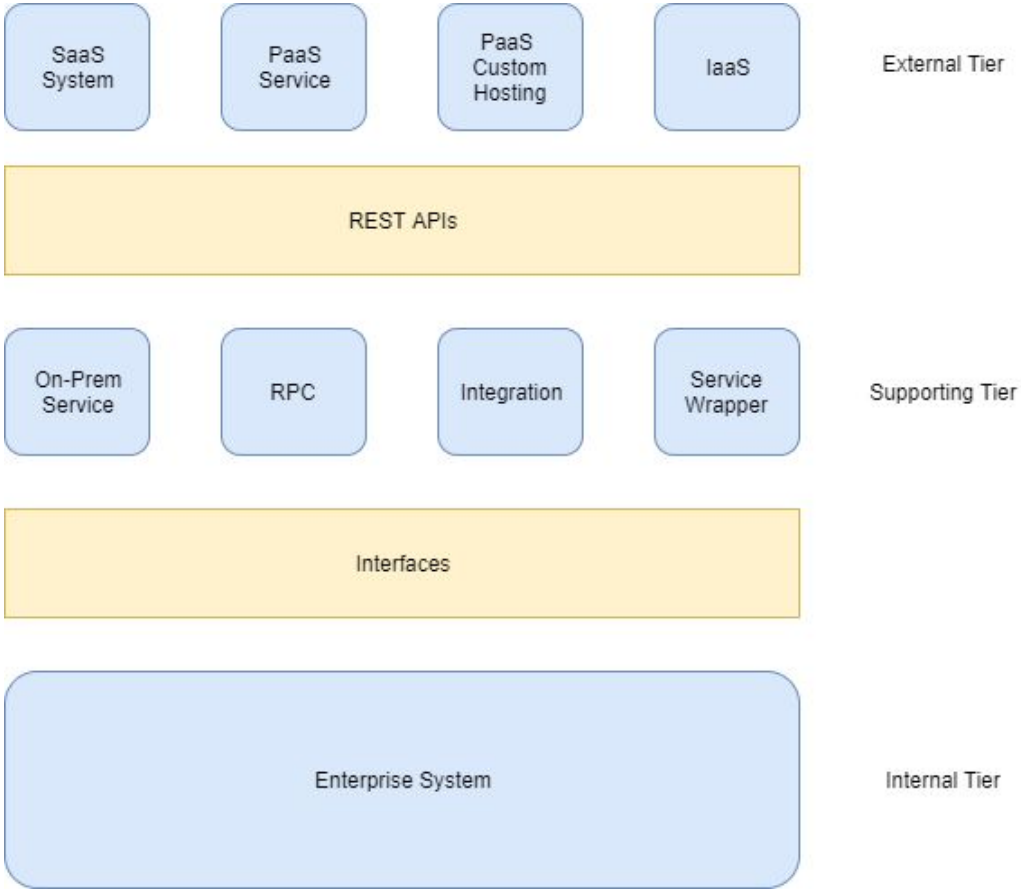

richard@tyregrim.se




Self Contained Unit Of Execution

# Bryt Monoliten




SaaS System
PaaS Service
PaaS Custom Hosting
IaaS
External Tier
REST APIs
On-Prem Service
RPC
Integration
Service Wrapper
Supporting Tier
Interfaces
Enterprise System
Internal Tier

PaaS/SaaS

- Lambda/Serverless-exekvering
- Pub/Sub
- FIFO
- API-Gateways/Proxying
- Containers
- Autoskalning
- REST-wrappers i interface-implementationer
- Flexibel lagring, passande mikrotjänster
- Köp eller utveckla funktionalitet med mycket hög granularitet, efter behov
- Distribuera ansvar

```
// This server can run on App Engine.
package main

import (
    "fmt"
    "log"
    "net/http"
    "os"
)

func main() {
    port := os.Getenv("PORT")
    if port == "" {
        port = "8080"
    }
    http.HandleFunc("/", hello)

    log.Fatal(http.ListenAndServe(fmt.Sprintf(":%s", port), nil))
}

func hello(w http.ResponseWriter, r *http.Request) {
    w.Write([]byte("Hello, 世界"))
}
```

```
gcloud app deploy
```

## Exempel App Engine/Go


Best of breed

- Skala horisontellt
- Små enheter, skalar snabbt
- Använd mönster för att trigga autoskalning
- Serverless PaaS för autoskalning
- Klustermodeller on-prem
- Docker
- Kubernetes

- Fail Fast
- Single-Responsibility
- Circuit Breaker
- API Gateway
- Worker Pool
- Domain Boundary Strategy
- Service Discovery Strategy
- Variable Schema Data
- Schemaless Data
- Event Driven
- Asynchronous Execution
- Strategy
- Decorator

# Switchable Strategies

```
Customer customer = new Customer();

// Injected part
customer.type = CustomerType.Gold;
ICalculator calc = CalculatorFactory.GetLoanCalculator(customer);
// ---

int score = calc.GetScore(customer);
int rc = calc.GetRecommendationClass(customer);
```

```
public class CalculatorFactory {

    public static ICalculator GetLoanCalculator(Customer cust) {
        if(cust.type == CustomerType.Regular) {
            return new SuperFinanceCalculator();
        }
        else if(cust.type == CustomerType.Gold) {
            return new SpecialCustCalculator();
        }
        else {
            return null; // Not really...
        }
    }

}
```

# Class Wrappers

```
public class WrapperMaxiCalculator implements ICalculator {
    private ICalculator basecalc; // The base object we are wrapping

    public WrapperMaxiCalculator(ICalculator basecalc) {
        this.basecalc = basecalc;
    }

    @Override
    public int GetScore(Customer cust) {

        int score = 0;

        // Build our request POJO
        CreditRequest req = new CreditRequest();
        req.Age = cust.age;
        req.AreaType = cust.area;
        req.Debt = cust.debt;
        req.Income = cust.income;
        req.NoLoans = cust.loans;

        // Make the call
        MaxiCreditScore maxiScore = MaxiServiceCaller.Call(req);

        // Align with legacy functionality, according to decided business rules (for now, then use additional capabilities when needed)
        score = ((maxiScore.instantPScore + maxiScore.longPSCore) / maxiScore.repayRate) - maxiScore.riskRating;

        // Add to existing score
        return basecalc.GetScore(cust) + score;


    }
```

# Class Wrappers

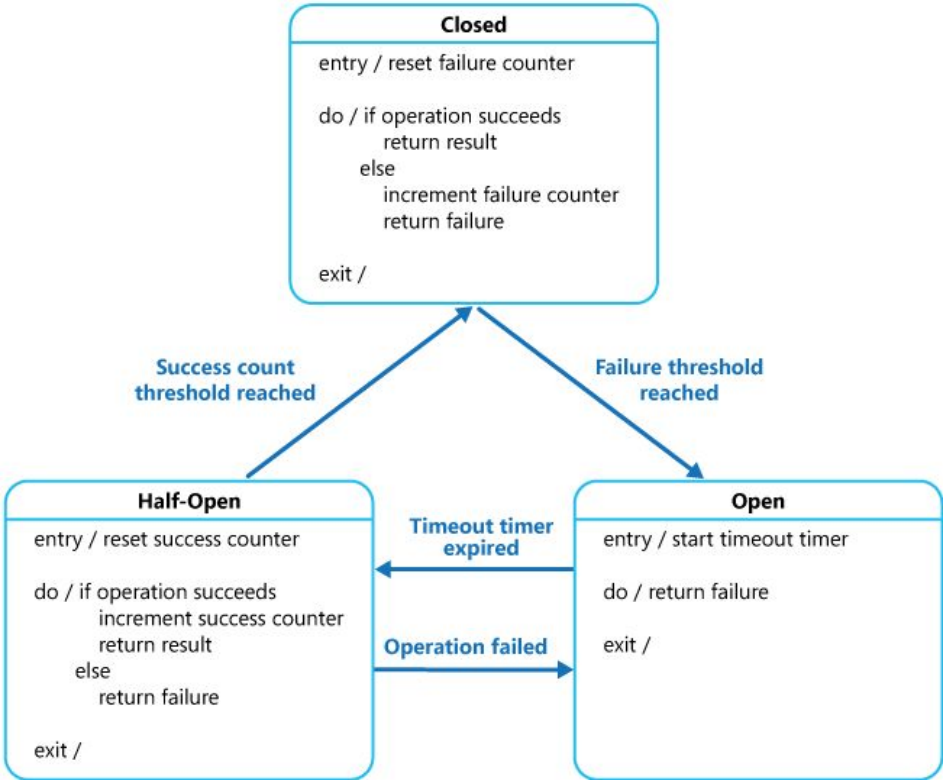
```
SuperFinanceCalculator calc = new SuperFinanceCalculator();

// ... Legacy Code ... //

// Injected part
WrapperMaxiCalculator extendedCalc = new WrapperMaxiCalculator(calc);

int score = extendedCalc.GetScore(customer);
```

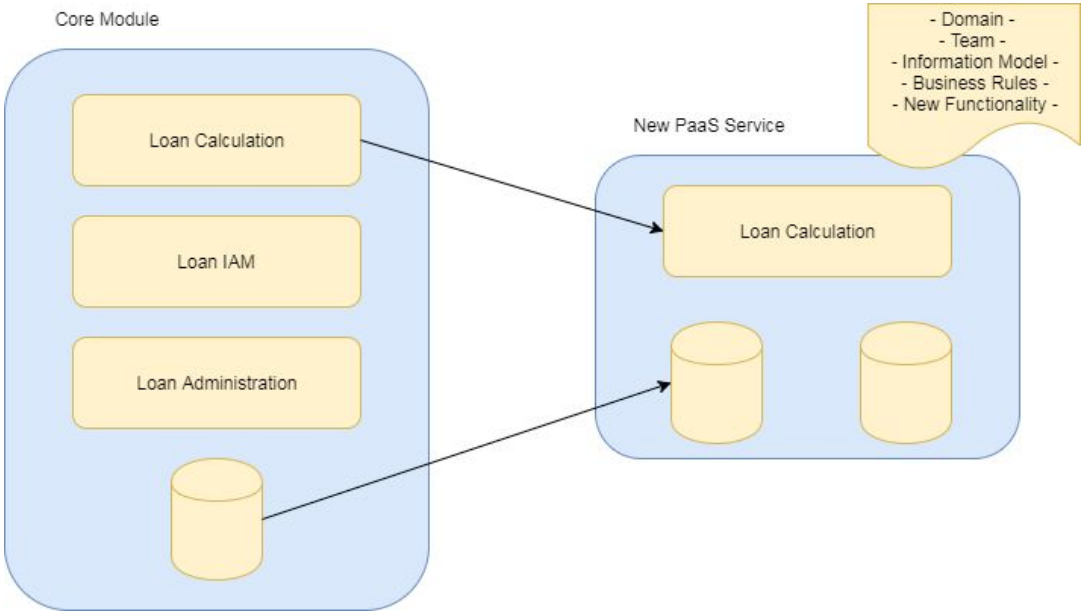
# Circuit Breaker




*Microsoft*

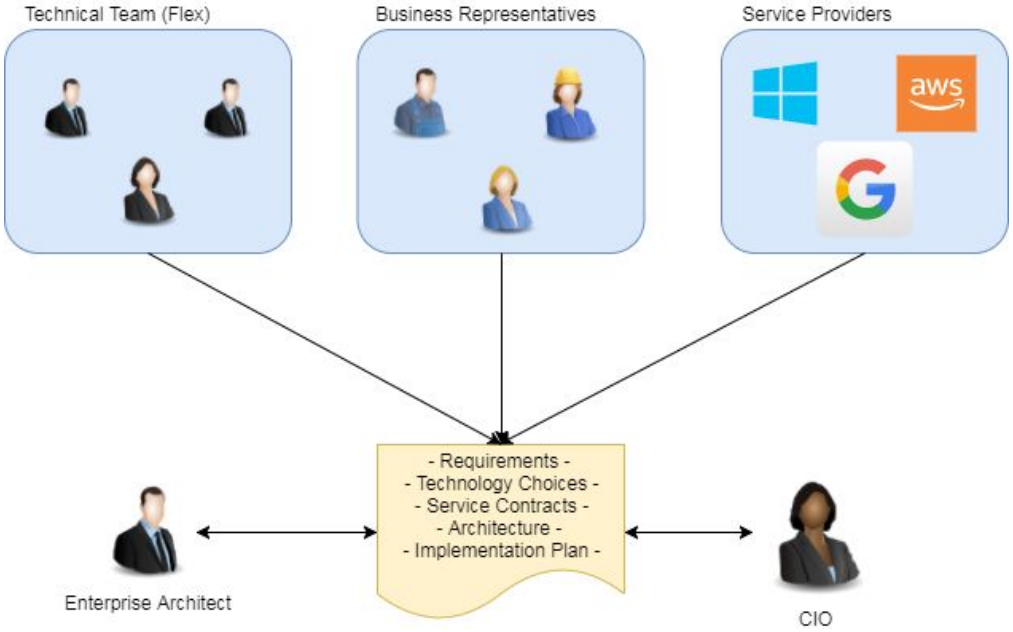
# Bryt ut en funktionalitet

- Identifiera isolerad funktionalitet
- Identifiera domän
- Bryt ut nödvändig data
- Datamodell
- Forma domänteam



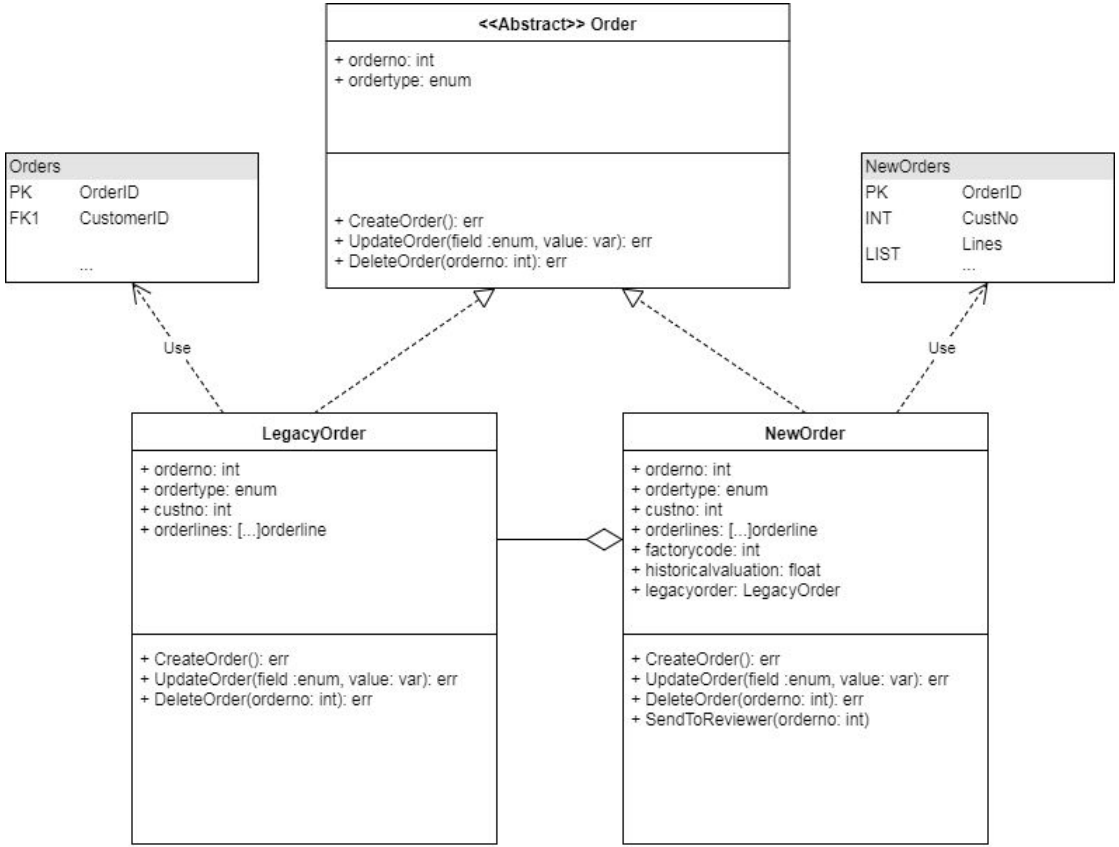
# Bygg en tjänst

- Arkitektur
- PaaS/SaaS eller on-prem
- Teknikval
- Plattformsval
- Teckna avtal för tjänster
- Custom eller helt PaaS/SaaS
- Bygg tjänst
- API/Service Proxying



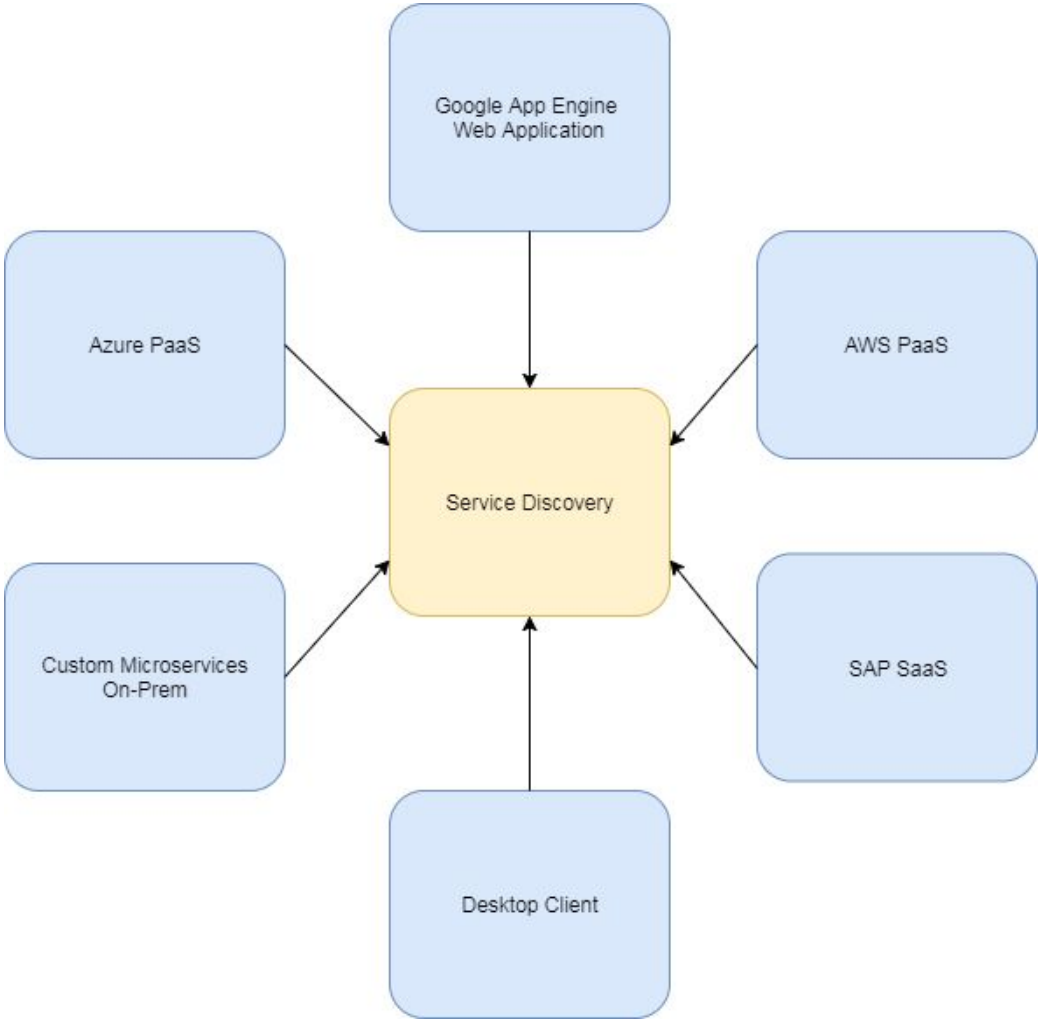
# Applicera mönster i befintlig kodbas

- Wrappers
- RPC
- SOLID!
- API Proxying
- Separera data
- Open/Closed Principle
- Inga förändringar i befintliga bibliotek
- Strangler Pattern



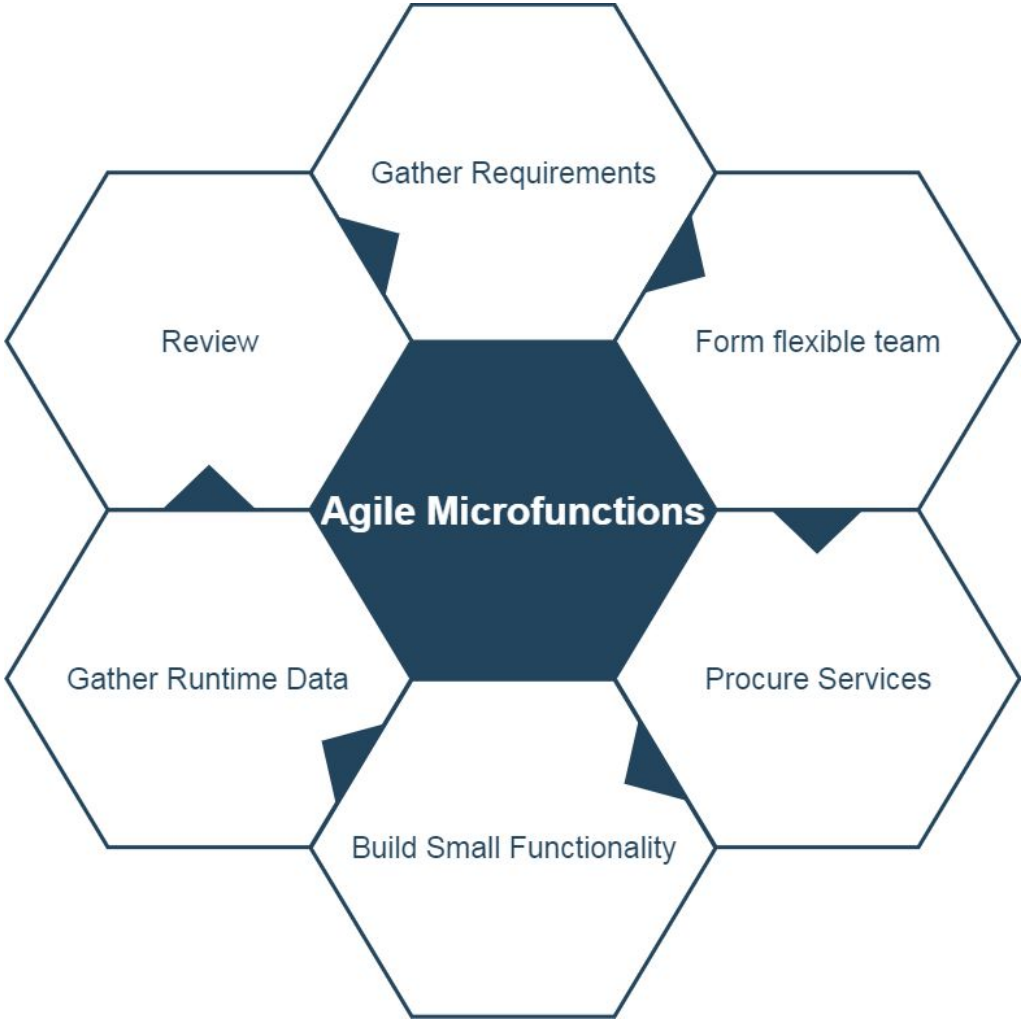
# Löst kopplad funktionalitet

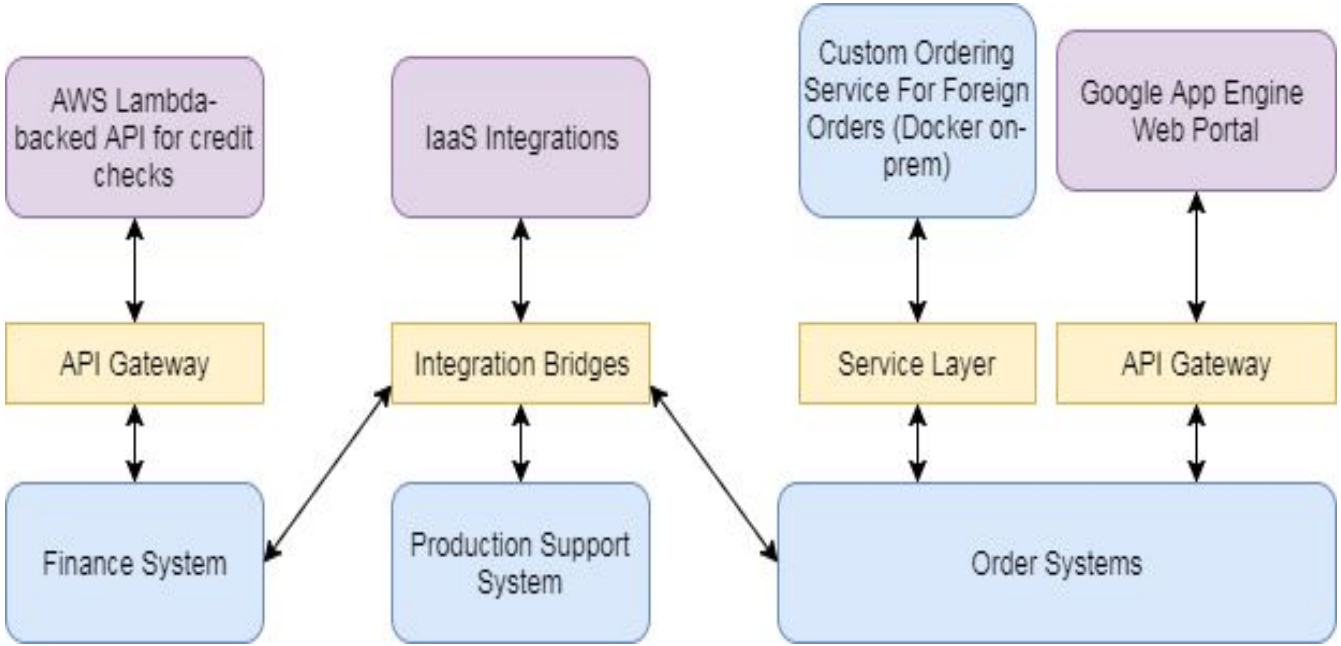
- Byt leverantör vid behov
- Best of breed
- Lösa avtal för tjänster
- Containers
- Återanvändbarhet
- Domänstyrt
- Behovsstyrt
- Flexibla team
- Konkurrensutsätt och omvärdera
- Enklare att underhålla
- Agilt



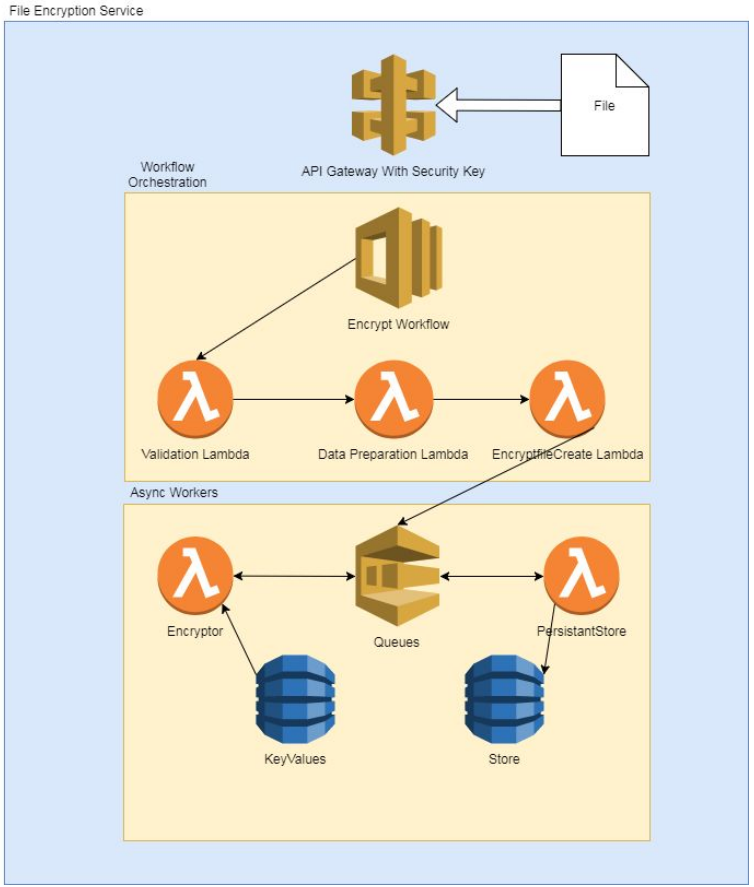
# Kontinuerlig Review

- Lätt att byta ut en enskild del
- Lätt att bygga om
- Byt plattform eller teknik vid behov, utan impact på övriga delar av systemen
- Korta avtal och abbonemangstjänster möjliggör snabba byten av PaaS-leveranser




AWS Lambda-backed API for credit checks
IaaS Integrations
Custom Ordering Service For Foreign Orders (Docker on-prem)
Google App Engine Web Portal
API Gateway
Integration Bridges
Service Layer
API Gateway
Finance System
Production Support System
Order Systems

Exempel


File Encryption Service
File
API Gateway With Security Key
Workflow Orchestration
Encrypt Workflow
Validation Lambda
Data Preparation Lambda
EncryptfileCreate Lambda
Async Workers
Encryptor
Queues
PersistantStore
KeyValues
Store

# Exempel