



COMPLEXITY GAME – FROM BIG BALLS OF MUD TO SHINY BULLETS

Raimondas Tijūnaitis

Adform

Established in DK, 2002

350+ engineers

7 data centers across 4 continents

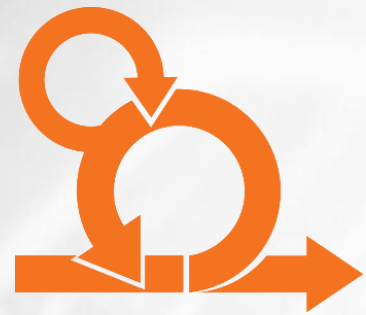
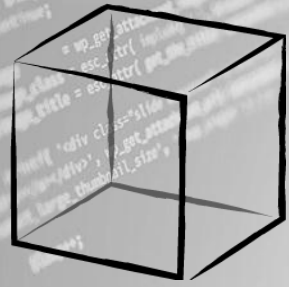
2 million requests/s in less than 100ms

50 billion transactions a day

3000+ servers

Petabytes of data

“Modern” software evolution



One and only one true metric

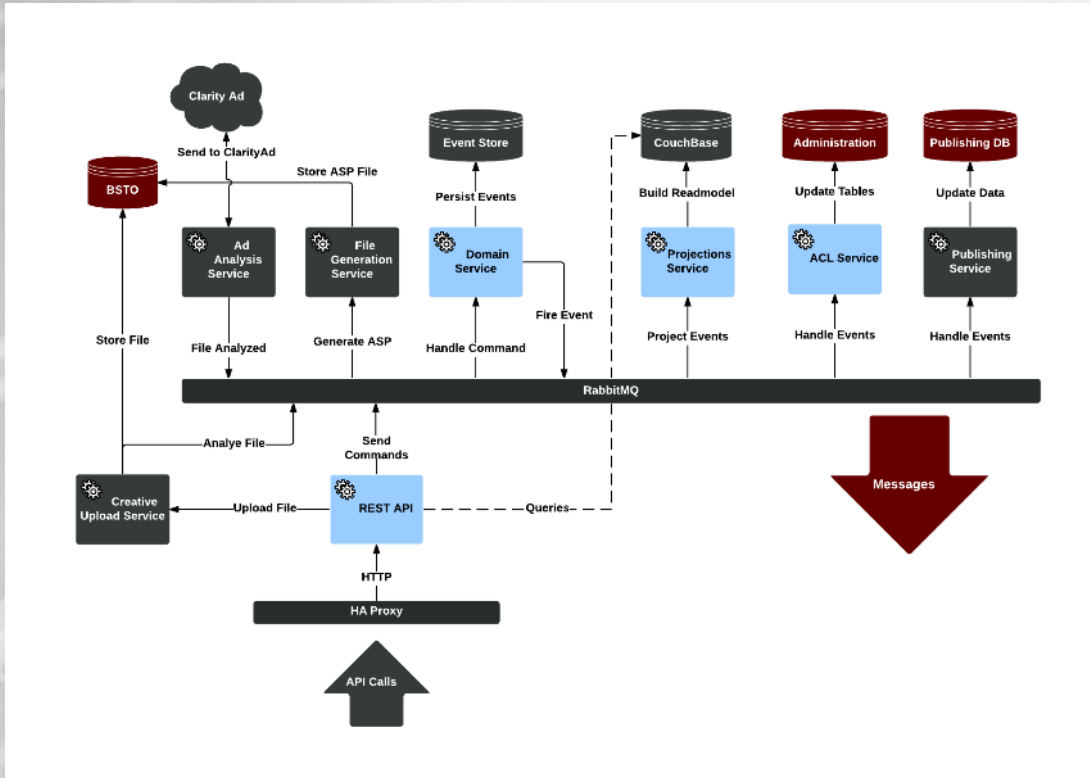
Is it harder to make changes?

Physics against us?

- A computer program that is used will be modified
- When a program is modified, its complexity will increase, provided that one does not actively **work against this**.
-- *Lehman M. M*

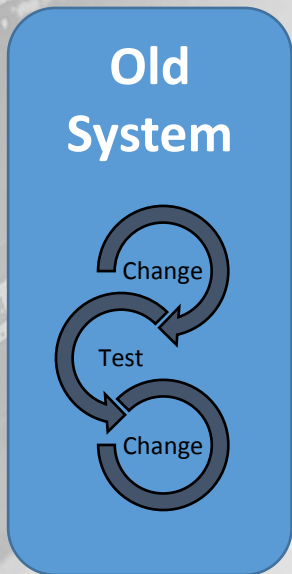


The princess, aka - Event-sourced micro-services

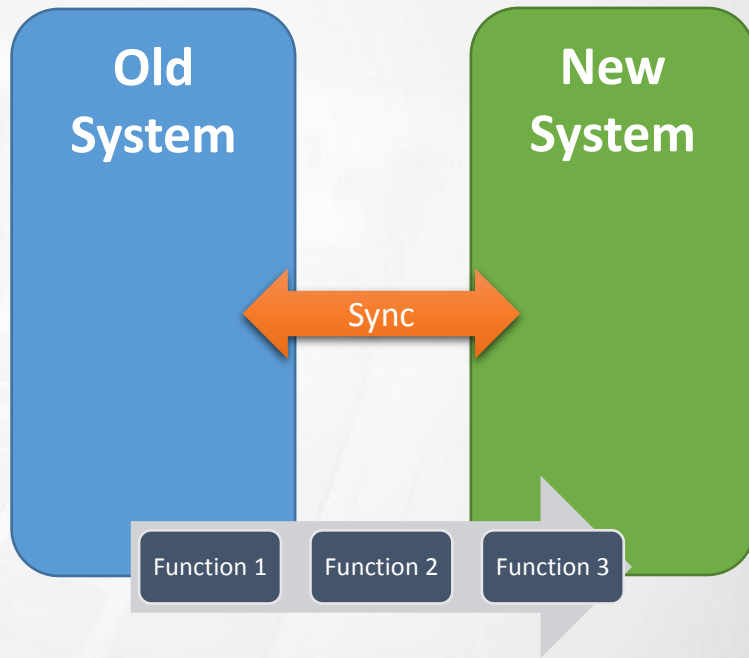


LEVEL 1: Picking the strategy

Refactoring



Rework



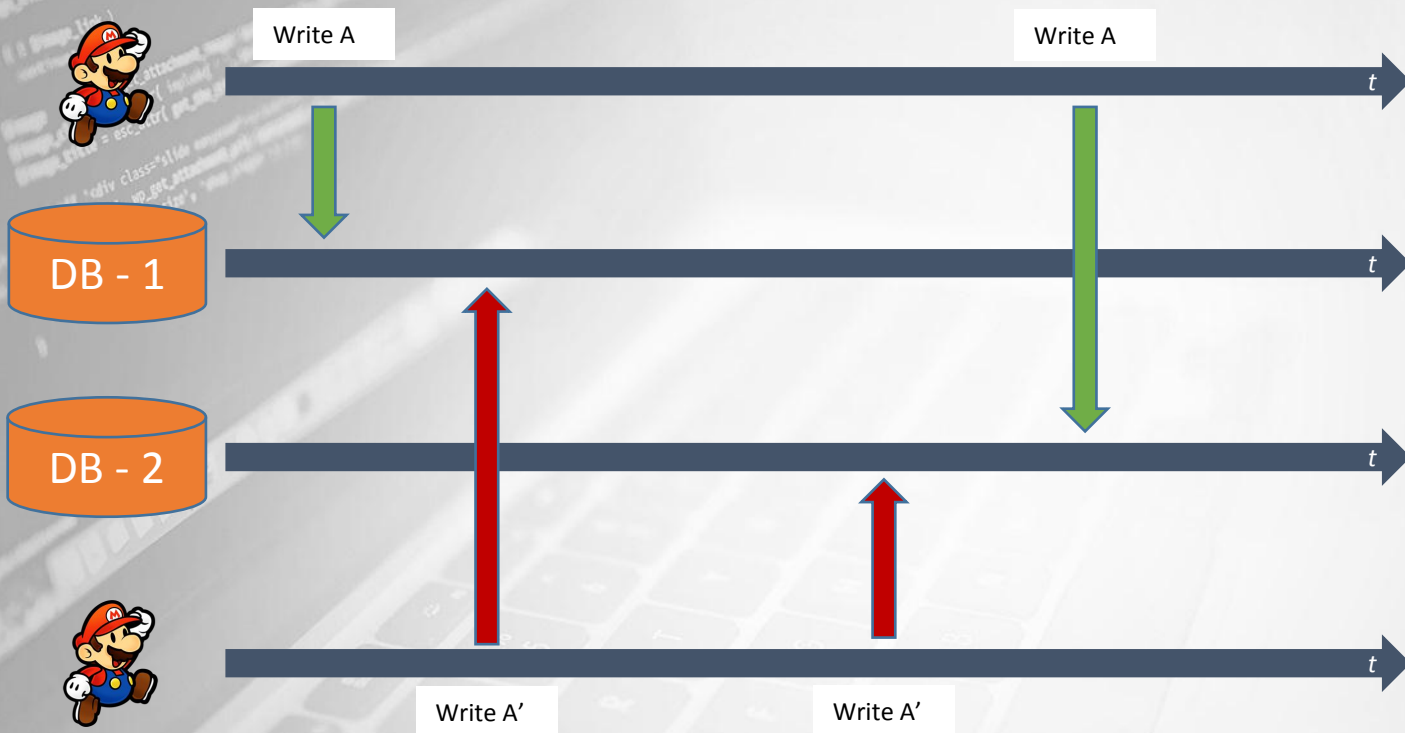
Messages from old system

```
public void MySuperAwesomeBusinessOperation()
{
    var transaction = new Transaction();
    transaction.Begin();

    try
    {
        //...
        Repository.Save(stuff);
        //...
        Events.Notify(new SomethingHappened(stuff));

        transaction.Commit();
    }
    catch (System.Exception)
    {
        transaction.Rollback();
        throw;
    }
}
```

Dual-writes: time and time again



Check!



Meet new enemies



Eventual consistency

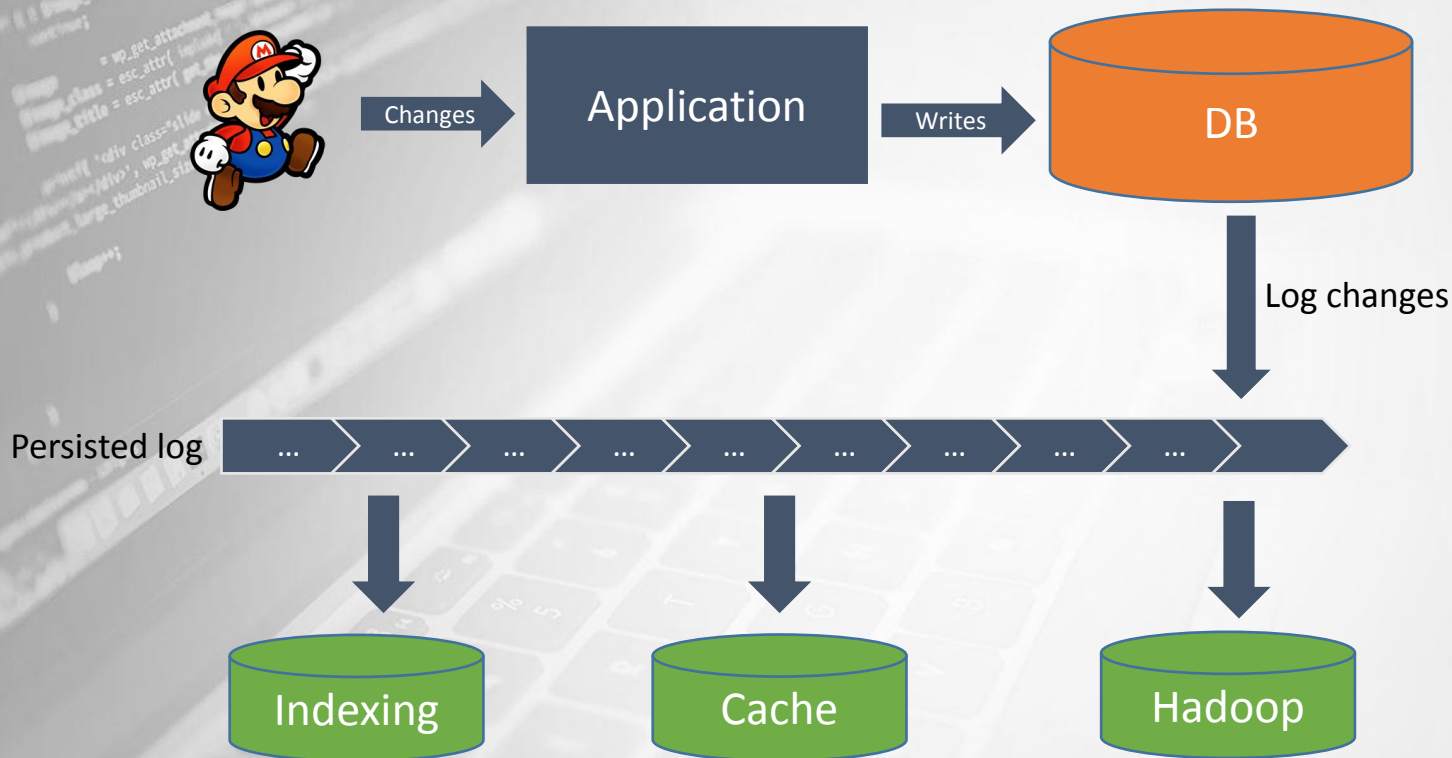


Idempotency and ordering



Desynchronization/dual-writes

SECRET LEVEL - CDCs



Technologies



samza



Use cases



Data replication



Building Caches and Secondary indexes



Data processing and transformation

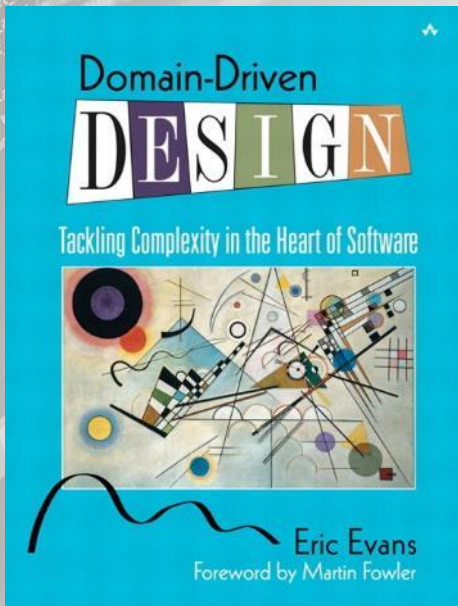


Building Read-models/Materialized views

Check!



LEVEL 2 – Strategic Design



Event Storming



Invite the right people

Explore the domain starting from Domain Events

Explore origin of the domain events – define commands

Check!



LEVEL 3: Event sourcing



Commands

Service

Events

Event Store

BannerCreated

BannerClickUrlsAdded

BannerRenamed

BannerRenamed

Event store

Stores series of immutable events

There is no delete

Acts as “Event Log”

Has built in projection engine

Is not designed for querying

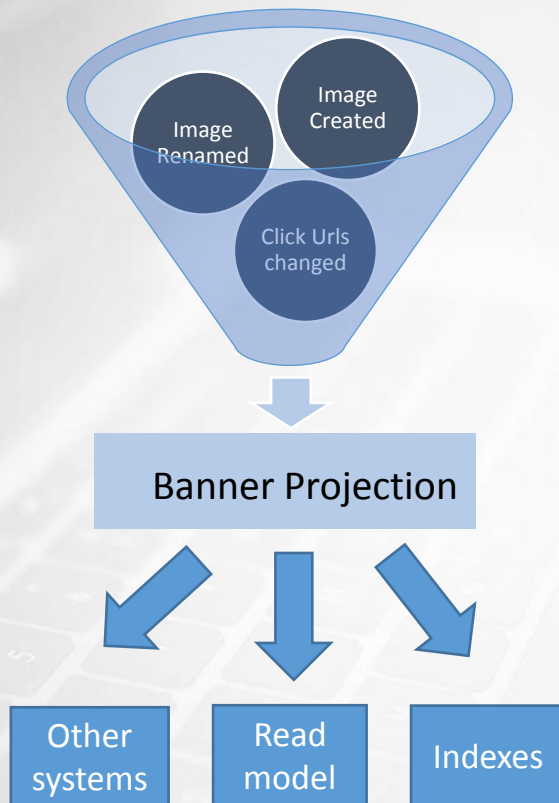


EVENT STORE.

Check!



Level 4: Projections



Check!



LEVEL 5: API

Should be the only way to interact with your system

UI is built on top of the API

Clear and agreed guidelines



Commands as HTTP resources

Re-upload File



PUT: <http://.../ImageBaner/{uuid}/File>

Rename Banner

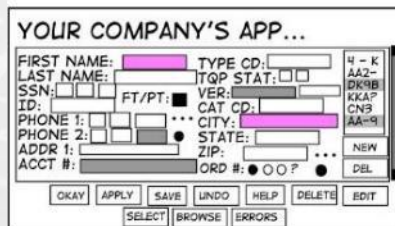
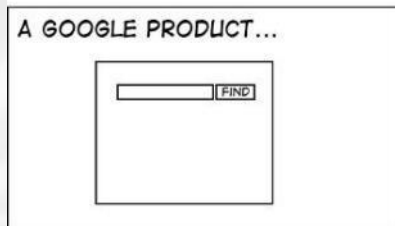
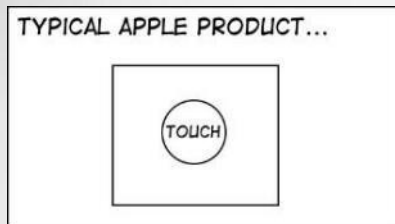


PUT: <http://.../ImageBaner/{uuid}/Name>

Check!



LEVEL 6: Task based UI



STUFFTHATHAPPENS.COM BY ERIC BURKE

From excel to business cases

Name ▲	Supplier	Active	
Super Fun Club Membership	ACME	<input checked="" type="checkbox"/>	Deactivate
Norwegian Salmon	ACME	<input type="checkbox"/>	
Giant Bean Bag	Walmart	<input checked="" type="checkbox"/>	Deactivate
Watermelons	Patata	<input checked="" type="checkbox"/>	Deactivate
TShirts			

Inventory Item

Name

Supplier

Description

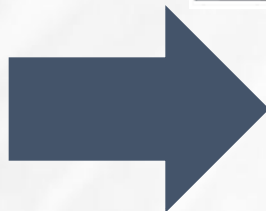
Supplier Cost

Count

Status ▼

Deactivation Comment

www.cqrsinfo.com



Deactivate Inventory Item

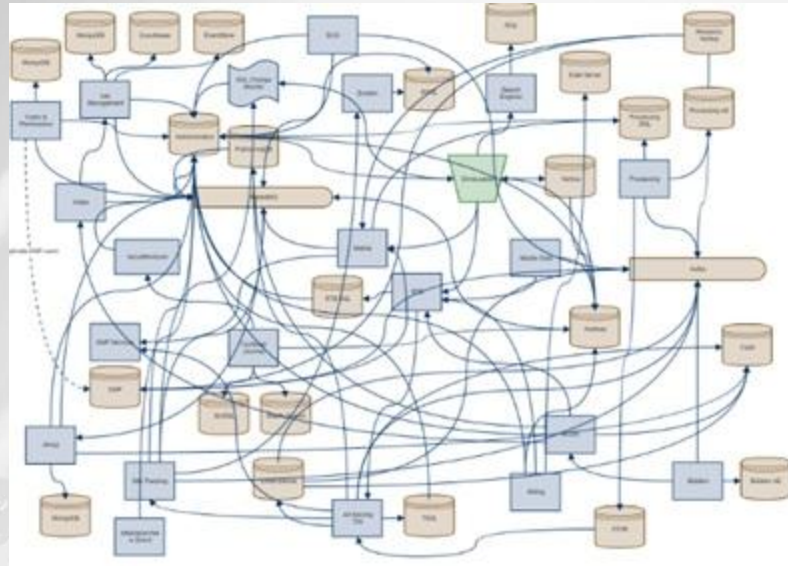
A comment is required explaining why you are deactivating the inventory item.

www.cqrsinfo.com

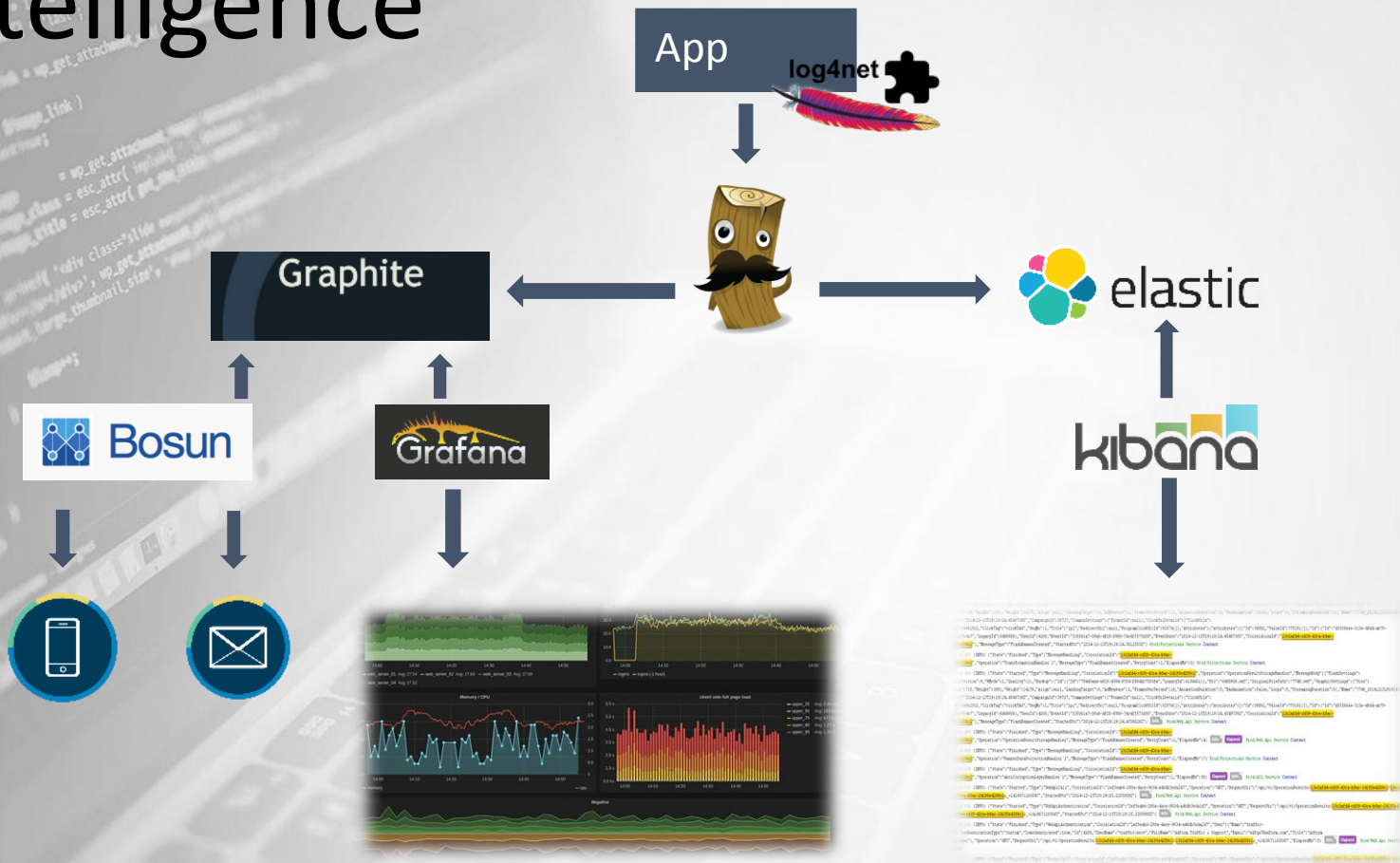
Check!



LEVEL 7: Distributed Monolith



Intelligence



Service toolkit libraries

Windows
Service

Data Sync
framework

Web API

Event-store
lib

Testing lib

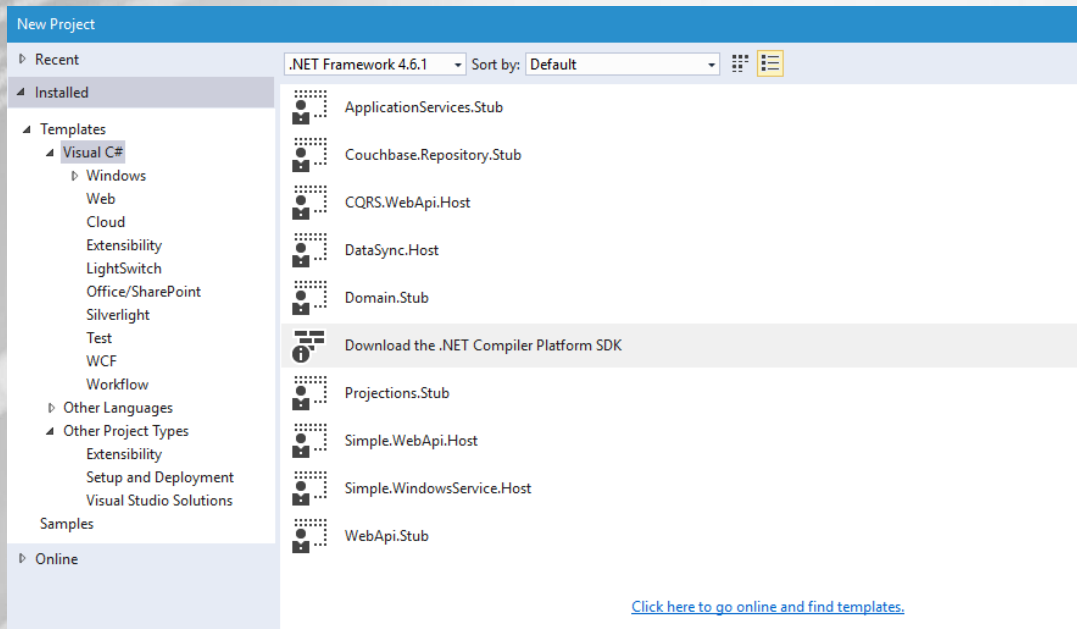
Caching lib

Pub/sub lib

Diagnostics
lib



Service Templates



The screenshot shows the 'New Project' dialog in Visual Studio. The left sidebar is expanded to 'Visual C#' > 'Web'. The main pane shows a list of templates for '.NET Framework 4.6.1'. The templates listed are:

- ApplicationServices.Stub
- Couchbase.Repository.Stub
- CQRS.WebApi.Host
- DataSync.Host
- Domain.Stub
- Download the .NET Compiler Platform SDK
- Projections.Stub
- Simple.WebApi.Host
- Simple.WindowsService.Host
- WebApi.Stub

At the bottom right of the dialog, there is a link: [Click here to go online and find templates.](#)

Check!



But what if...



Adapt to business thinking

Avoid
technical
arguments

Code quality is poor

We need to refactor

Technology is old

Everyone is using Angular now...

Missing “best practices”

We want to try NoSQL...

Prefer
economical
arguments

Releases every day

Faster feature development

New products

Involve more developers

MS licenses are expensive...

Don't get lost in a dungeon

Have a **VISION**

COMMIT

Define Your **STRATEGY**

Create **TACTICS**

FIGHT

ADOPT new tactics

Fight **HARDER**

WIN



Thanks



.NET

DEVELOPER

DAYS

**net.developerdays.pl
[@DeveloperDaysPL](https://twitter.com/DeveloperDaysPL)**