# SIMPLE MADE EASY

## *A Software Silver Bullet?*

PYCON CANADA
NOV 13, 2016

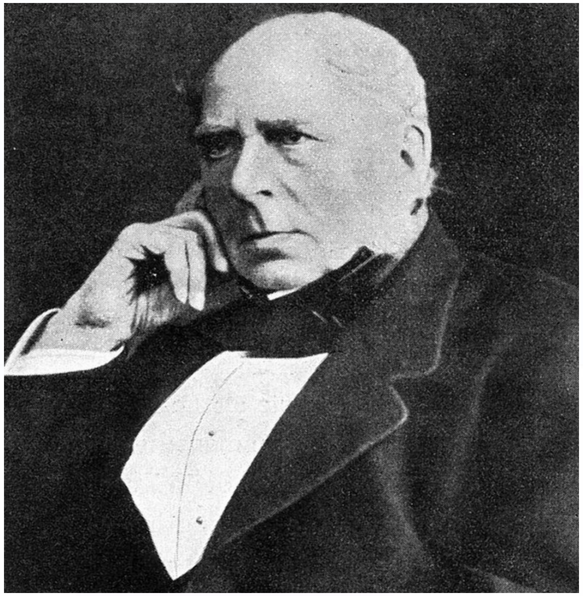
**Steve Jackson**

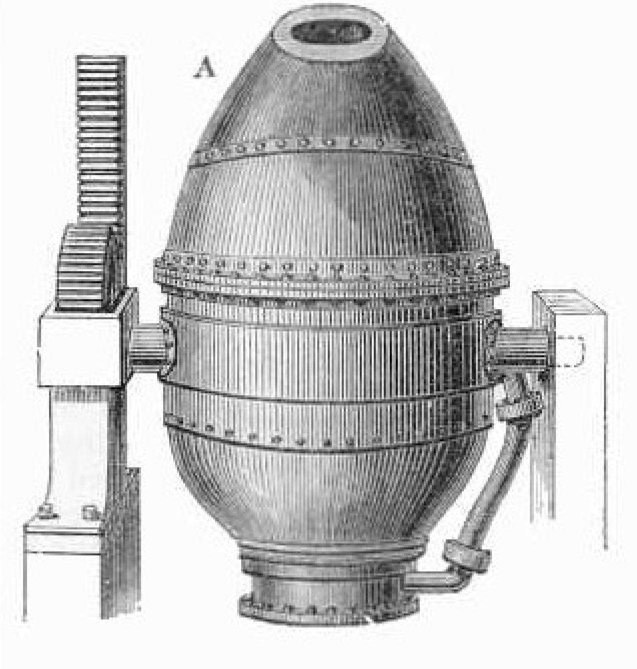
Software has been ***IN CRISIS***-

*for 30 years!*

*Because it is too hard to do a good job!*

# In 1850, STEEL was IN CRISIS!





The **Bessemer Process** was the first inexpensive industrial process for the mass-production of steel from molten pig iron.

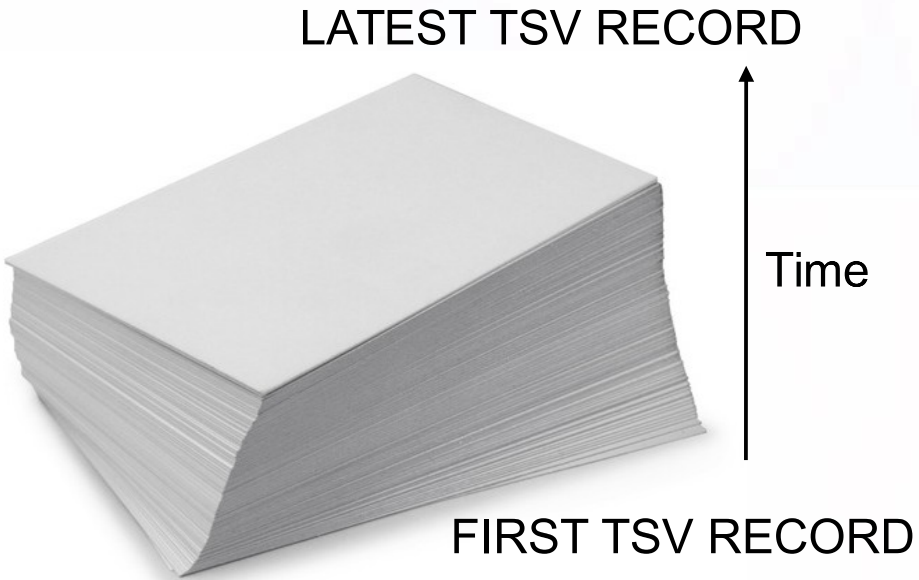
## Software Silver Bullet recipe:

| | | |
|---|---|---|
| 1. Data | ... | DS Records are Variable length TSV's |
| 2. Persistence | ... | Immutable TSV stores |
| 3. Logic | ... | Microservices |
| 4. Messaging | ... | Rabbit MQ (AMQP, Erlang) |

# IMMUTABLE Data storage – Keeping Data Simple!

CRUD = Create, Read, Update, Delete

IMMUTABLE = Create, Read.



You can Create an Update of an old record.
The Banks do this.

## Silver Bullet Microservices:

1. Small scope and mission. A single-person program.

2. Easy to comprehend and maintain.

3. The sole interface is Publishing & Subscribing DS Records.

4. Can use any Database internally. But you cannot expose it.

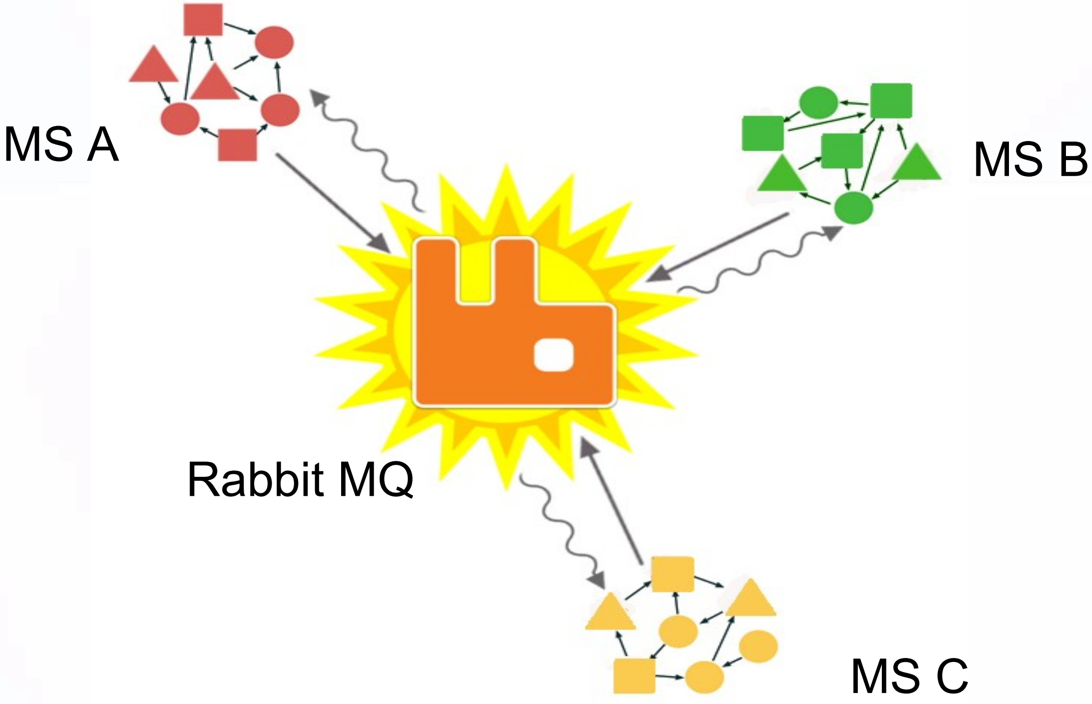
5. Mission is well defined and documented.

6. Might have a GUI, or be dark.

# HIGH Cohesion & LOW Coupling

High Cohesion means keeping the related parts of a code base in a single place.

Low Coupling is about separating unrelated parts of the code base as much as possible.



# Let's design a Hotel ERP System!

Start with **Room Service:**

1. Define a Microservice that formats and Publishes a **`Food Order`** Record.

2. Define a Kitchen MS that Subscribes to **`Food Order`** Records.
   When the food is ready, it Publishes a **`Room Delivery`** record.

3. Define a Room Service Waiter MS that Subscribes to **`Room Delivery`** records.

4. Define a Waiter MS app that Publishes a **`Delivery Complete`** record.

5. Define a Front Desk Billing MS that Subscribes to **`Delivery Complete`**, and bills the room.

We have a delightful ***Separation of Concerns*** between the MS.

Now let's do **Room Cleaning:**

1. Define a Front Desk MS that formats and Publishes a `Client Checkout` DS Record.

2. Define a Cleaner Dispatch MS that Subscribes to `Client Checkout` records. When a Cleaner is available it Publishes a `Room Clean Order` record.

3. Define a Cleaner MS that Subscribes to `Room Clean Order` records. When the room is clean, it Publishes a `Room Available` record.

4. Enhance the Front Desk MS to Subscribe to `Room Available` records. So it can maintain a database of available rooms.

Again, a delightful ***Separation of Concerns*** between the MS.

## You can Wash and Repeat this design process!

1. Room Reservation functionality

2. Human Resources functions, to schedule the room cleaners, waiters, etc.

3. Repair and Maintenance functionality

4. Etc.

Always, a delightful ***Separation of Concerns*** between the MS.

# A FEW BENEFITS OF DSI

1. Inherently Agile. Late-arriving change orders are welcome, not disruptive.

2. No formal requirements analysis phase – incremental requirements

3. You can develop MS in parallel

4. Can re-create bug conditions from the Immutable archives!.

5. Assured disaster recovery, from the Immutable archives!

6. DS Records can be analyzed for any Business Intelligence purpose in the future.

SteveJ@I-Technologyinc.com