



cloud
strategy
day

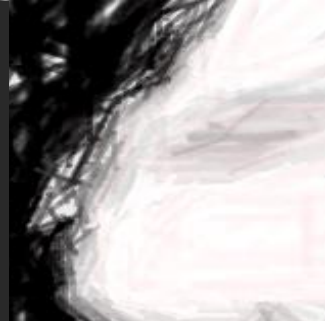


“How the Microsoft cloud enables the best App experiences across devices”

Beat Schwegler – beatsch@microsoft.com
Director, Platform Strategy Group, Microsoft Corp.


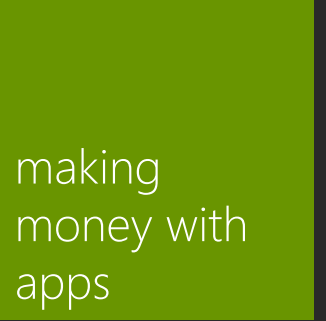
agenda



Microsoft's
devices and
services
strategy



making
money with
apps



cloud enabled
app scenarios
with Windows
Azure



key
architectural
considerations

chapter IV



service
architecture



data
partitioning



multitenancy



DevOps



integration





service
architecture

data
partitioning

multitenancy

storage

integration

service architecture

Device Client/Browser

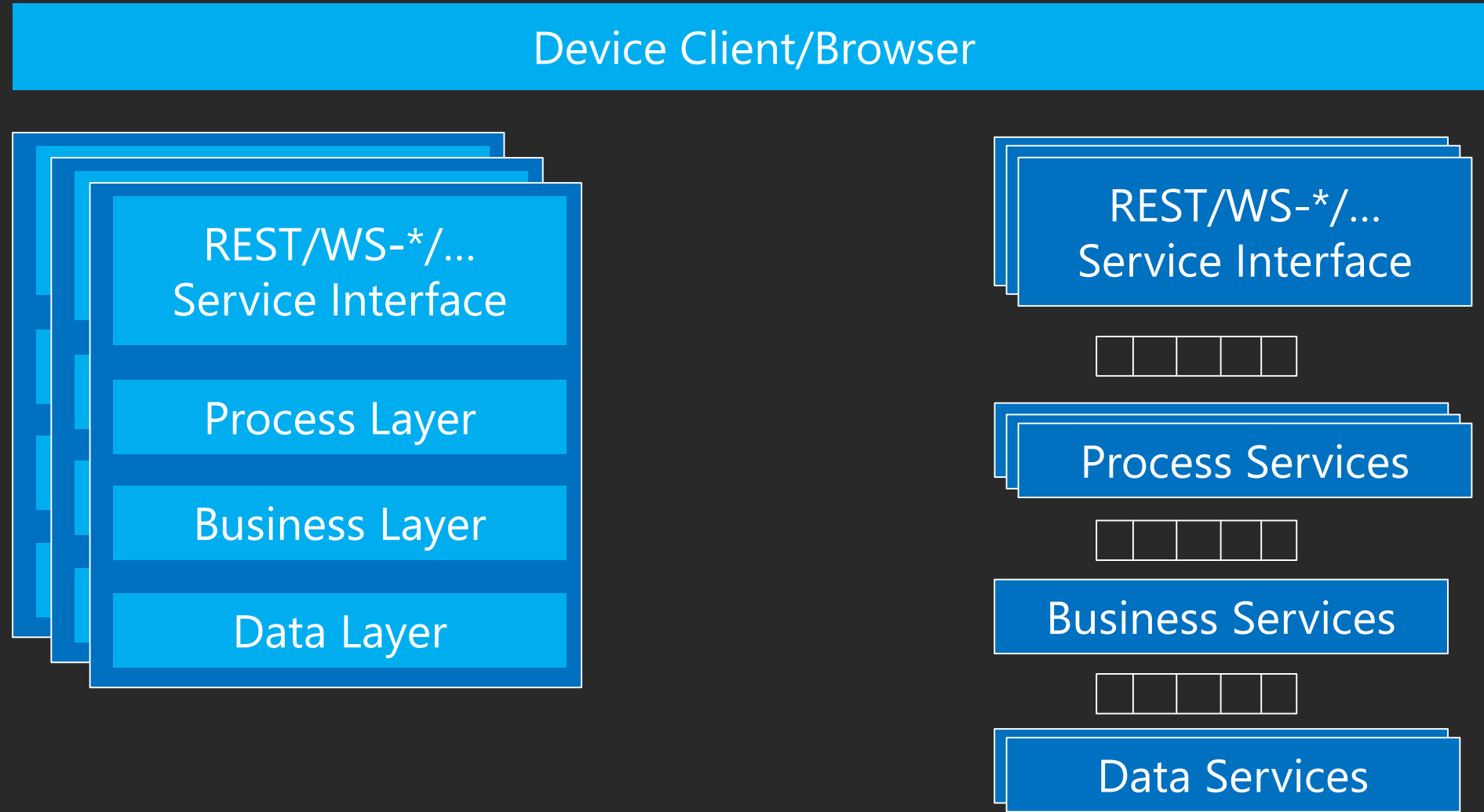
REST/WS-*/...
Service Interface

Process Layer

Business Layer

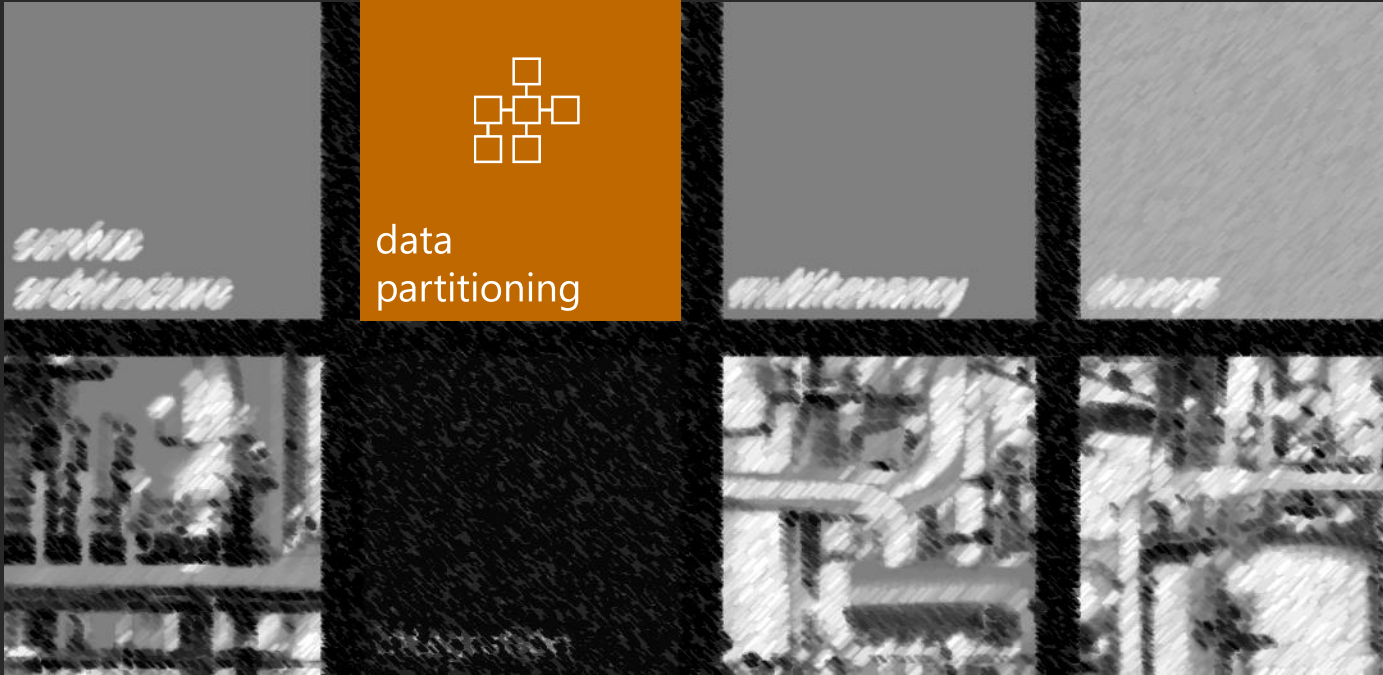
Data Layer

service architecture

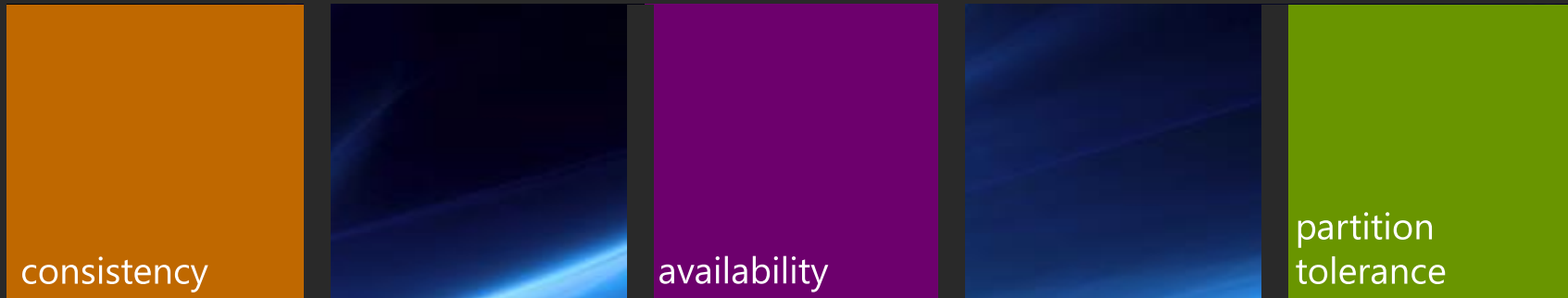


each instance contains all layers
inproc communication

instance per layer, communication through queues



cap theorem



data partitioning



traditional
reasons

data volume (too many bytes)

work load (too many
transactions/second)

new 'cloud
focused' reasons


cost (using different cost storage)

elasticity (just in time partitioning
for high load periods)



horizontal partitioning

First Name	Last Name	Email	Thumbnail	Photo
David	Alexander	davida@contoso.com	3kb	3MB
Jared	Carlson	jaredc@contoso.com	3kb	3MB
Sue	Charles	suec@contoso.com	3kb	3MB
Simon	Mitchel	simonm@contoso.com	3kb	3MB
Richard	Zeng	richardz@contoso.com	3kb	3MB



The diagram illustrates horizontal partitioning of a table. The table is divided into four horizontal sections, each representing a partition. Each partition is color-coded and contains one or more rows. Arrows point from the 'First Name' column of each row to a server icon below, indicating that the data is distributed across multiple servers. The servers are represented by a blue server rack and a blue database cylinder.

- Partition 1 (Blue border): David Alexander (davida@contoso.com), 3kb Thumbnail, 3MB Photo. Points to Server 1.
- Partition 2 (Orange border): Jared Carlson (jaredc@contoso.com), 3kb Thumbnail, 3MB Photo; Sue Charles (suec@contoso.com), 3kb Thumbnail, 3MB Photo. Points to Server 2.
- Partition 3 (Purple border): Simon Mitchel (simonm@contoso.com), 3kb Thumbnail, 3MB Photo. Points to Server 3.
- Partition 4 (Red border): Richard Zeng (richardz@contoso.com), 3kb Thumbnail, 3MB Photo. Points to Server 4.

horizontal partitioning (sharding)



spread data across similar nodes

achieve massive scale out (data and load)



intra-partition queries are simple

cross-partition queries are harder



vertical partitioning

First Name	Last Name	Email	Thumbnail	Photo
David	Alexander	davida@contoso.com	3kb	3MB
Jared	Carlson	jaredc@contoso.com	3kb	3MB
Sue	Charles	suec@contoso.com	3kb	3MB
Simon	Mitchel	simonm@contoso.com	3kb	3MB
Richard	Zeng	richardz@contoso.com	3kb	3MB



SQL Azure



Tables



BLOBS

vertical partitioning



place frequently
queried data in more
'expensive' indexed
storage



retrieving
a whole row requires
>1 query

spread data across
dis-similar nodes



place large data in
'cheap' binary
storage



hybrid partitioning

First Name	Last Name	Email	Thumbnail	Photo
David	Alexander	davida@contoso.com	3kb	3MB
Jared	Carlson	jaredc@contoso.com	3kb	3MB
Sue	Charles	suec@contoso.com	3kb	3MB
Simon	Mitchel	simonm@contoso.com	3kb	3MB
Richard	Zeng	richardz@contoso.com	3kb	3MB

The diagram illustrates hybrid partitioning. The table is partitioned by Last Name. The 'First Name', 'Last Name', and 'Email' columns are grouped into two partitions: one for 'Alexander', 'Carlson', and 'Charles' (blue border), and another for 'Mitchel' and 'Zeng' (orange border). The 'Thumbnail' column is grouped into one partition (purple border), and the 'Photo' column is grouped into another partition (red border). Arrows indicate that the 'First Name', 'Last Name', and 'Email' partitions are stored in two separate database instances (server and disk icons), while the 'Thumbnail' and 'Photo' partitions are stored in two separate database instances.

tables != rdbms

cross partition
queries are
resource
intensive



aggressive data
duplication can
save money
and boost
performance

storage is
cheap



goal: To be able
to include
Partition Key in
all queries

e.g. tweet storage

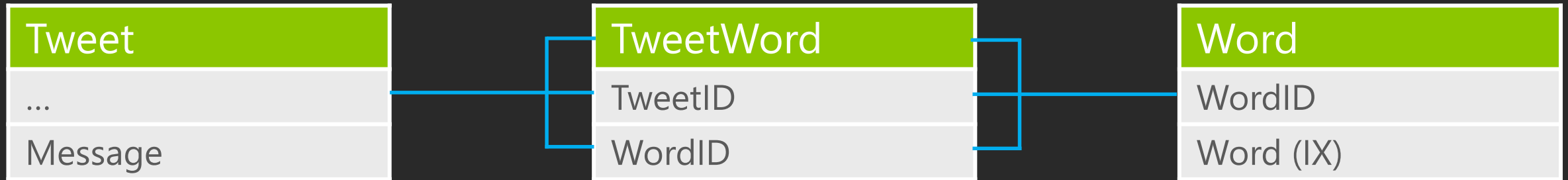
Tweet
TweetID
UserID
DateTimeStamp
Message

With an RDBMS you'd probably start something like this:

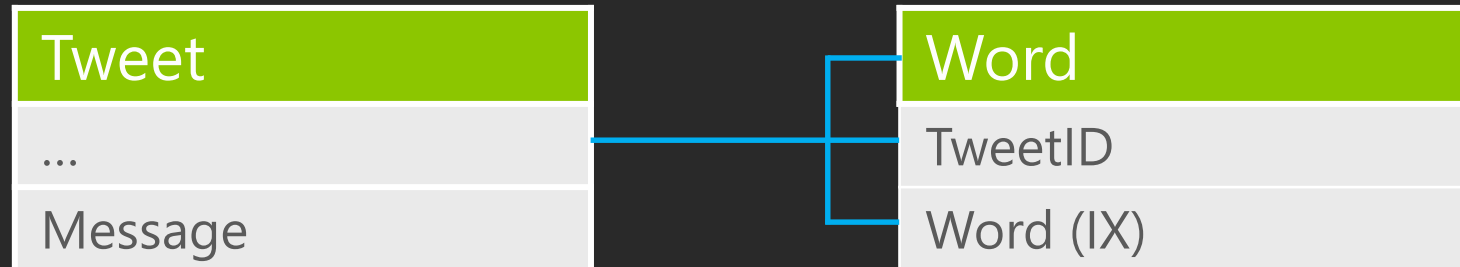
```
SELECT * FROM Tweet WHERE  
Message Like %SearchTerm%
```

e.g. tweet storage

You'd soon realize that LIKE isn't so wonderful.
You'd do a little normalization



Which quickly becomes this as $\text{len}(\text{key})$ approaches $\text{AVG len}(\text{word})$



e.g. tweet storage

With Tables we go the whole way

Tweet
TweetID (RK)
UserID (PK)
DateTimeStamp
Message

Worker Role Creates

TweetIndex
TweetID (RK)
UserID
DateTimeStamp
Message
Word (PK)

GET All Entities in Partition 'DavidA' from Tweet
GET All Entities in Partition 'Foo' from TweetIndex

e.g. tweet storage

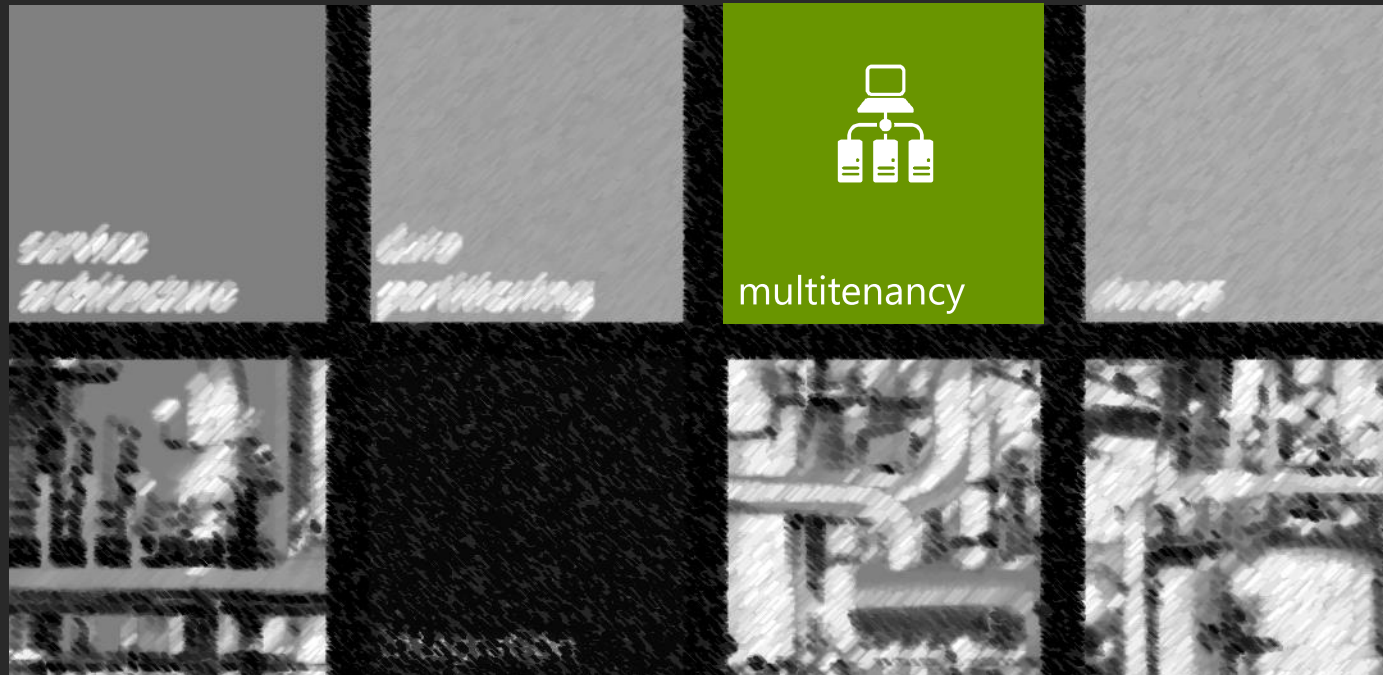
We may create multiple indexes

Tweet
TweetID (RK)
UserID (PK)
DateTimeStamp
Message

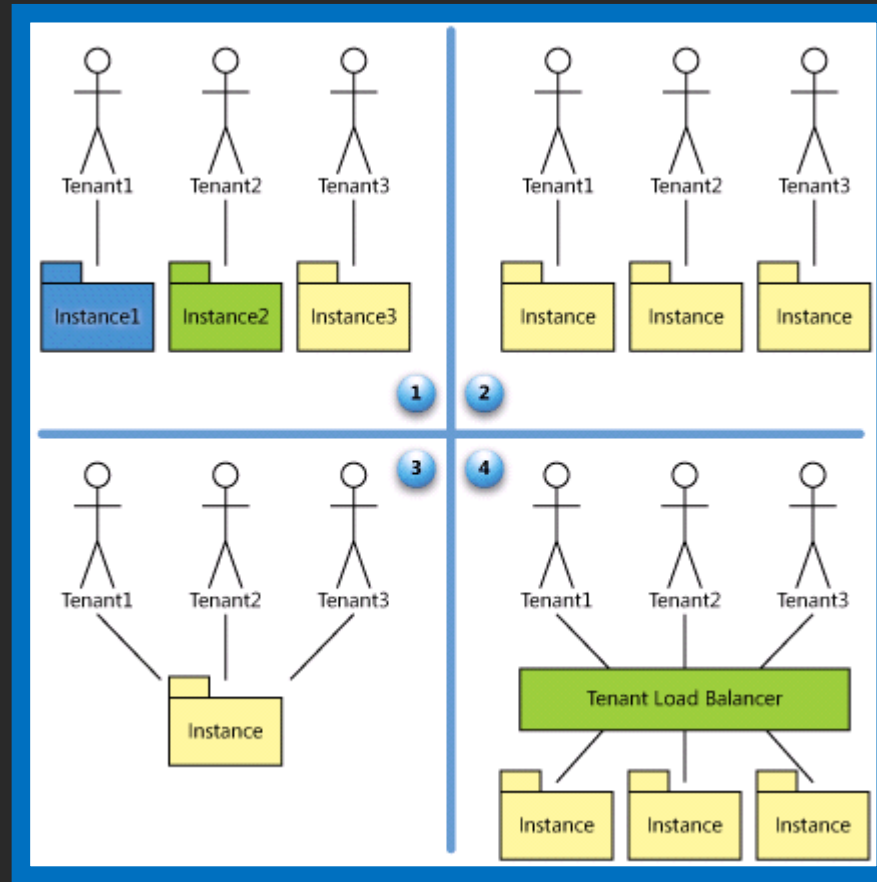
Worker Role Creates

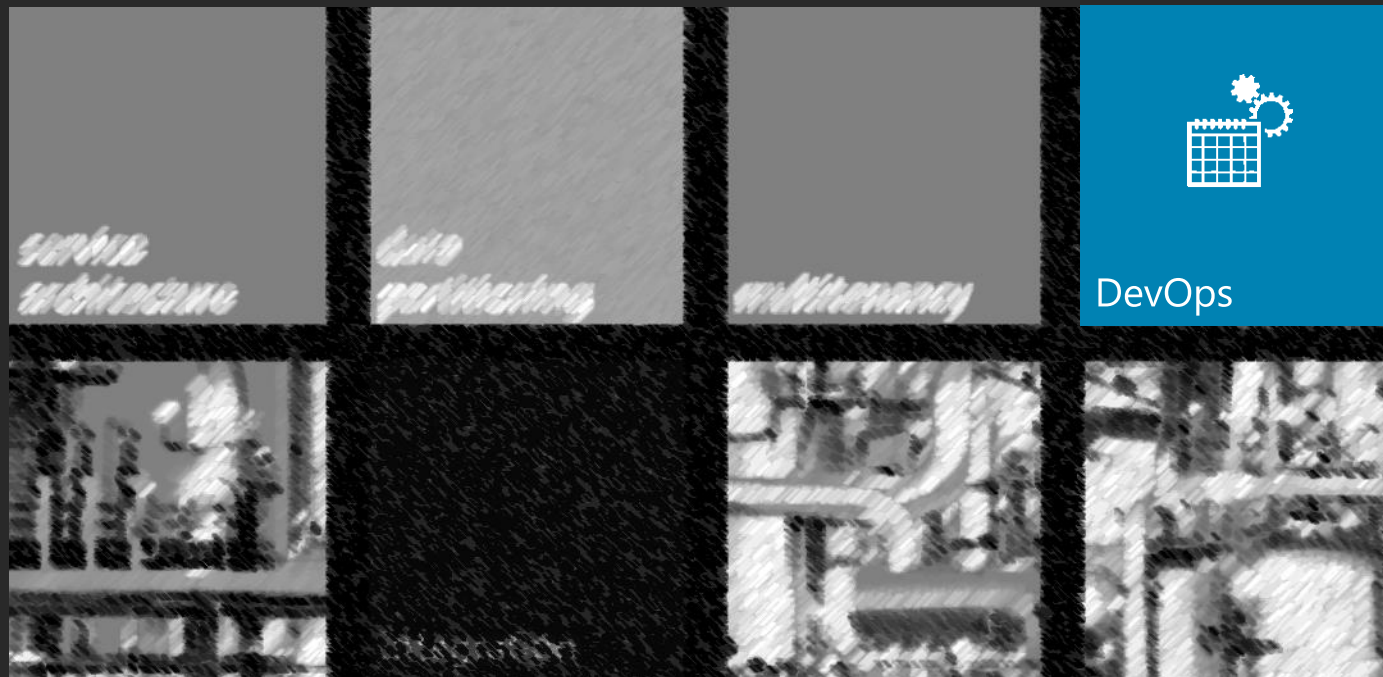
MentionIndex
TweetID (RK)
UserID
DateTimeStamp
Message
MentionedUserID (PK)

GET All Entities in Partition 'DavidA' from MentionIndex




multitenancy





continues deployment

FEATURING



Flickr was last deployed 2 days ago, including 4 changes by 1 person.

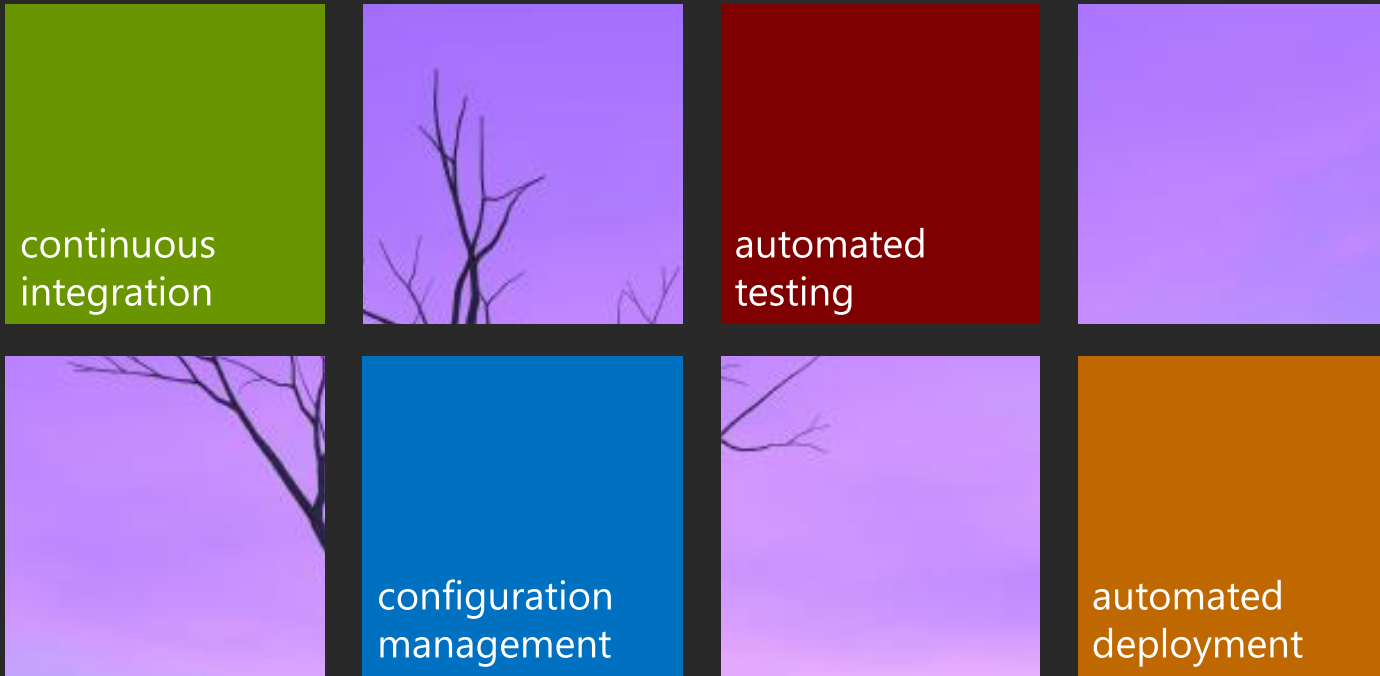
In the last week there were 77 deploys of 440 changes by 18 people.

<http://code.flickr.com/>

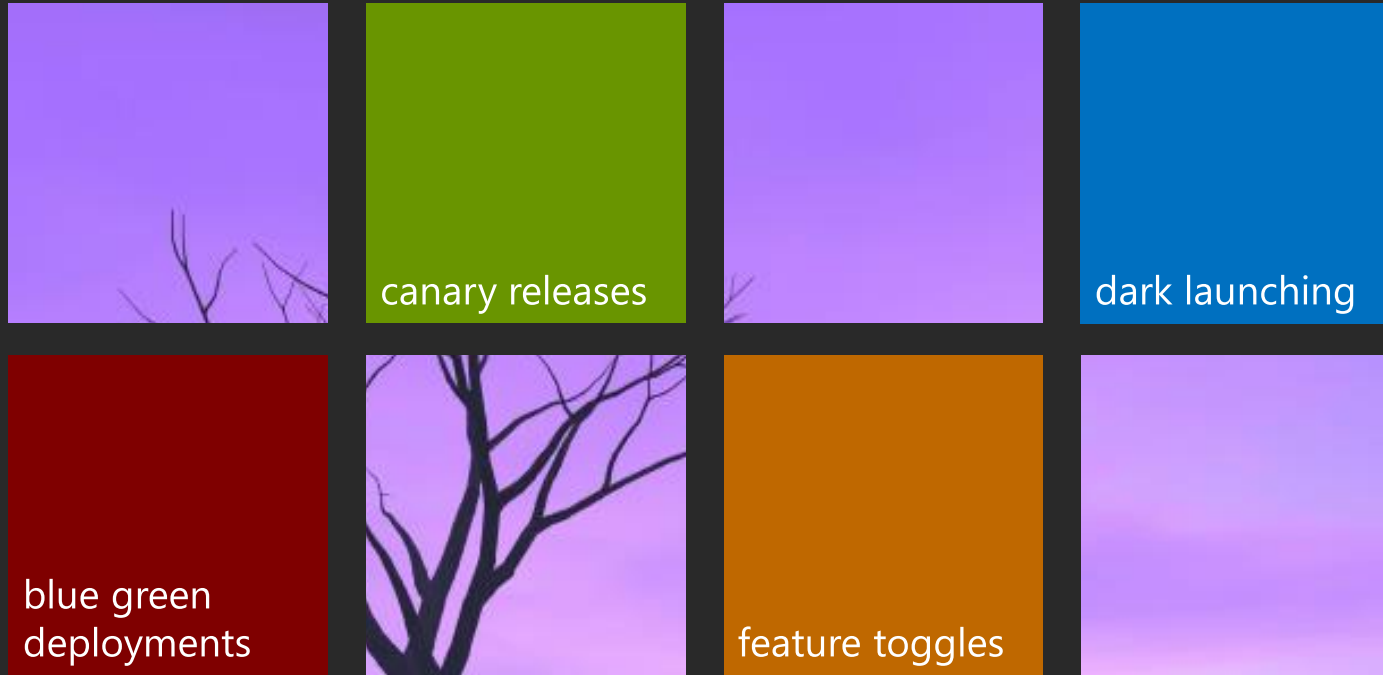
devops



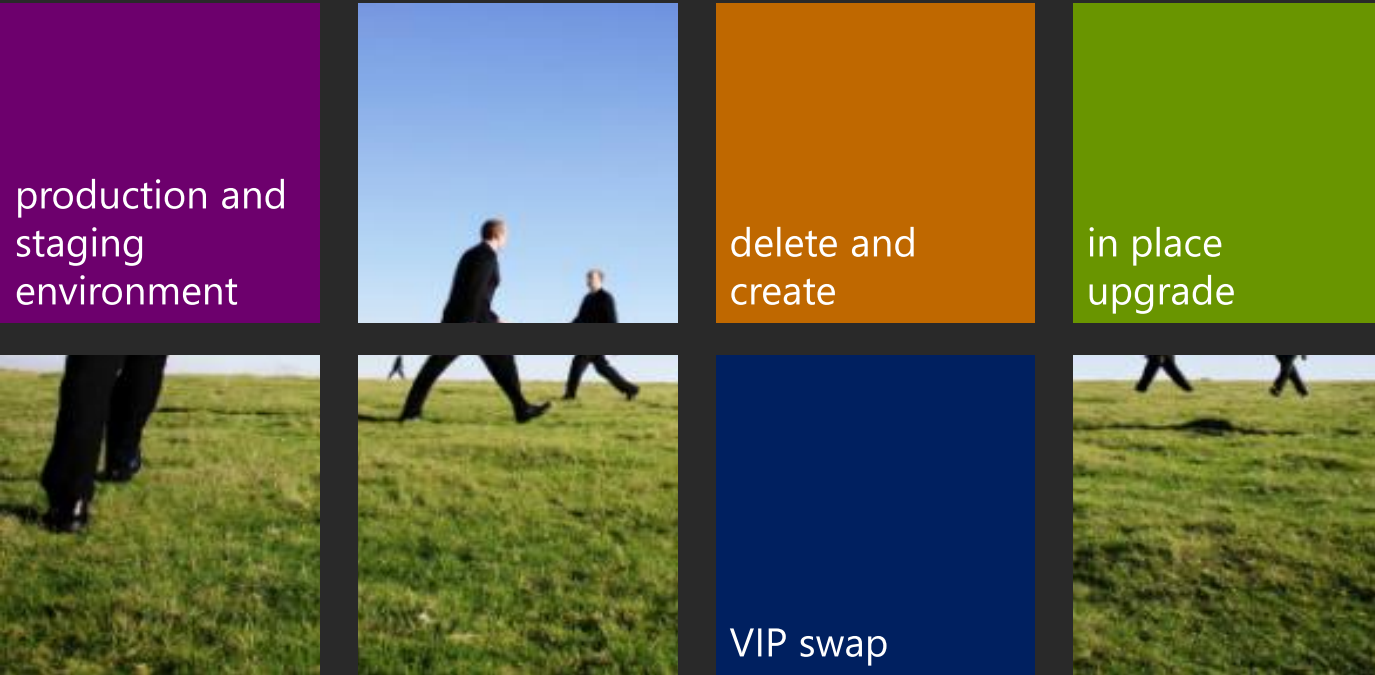
devops



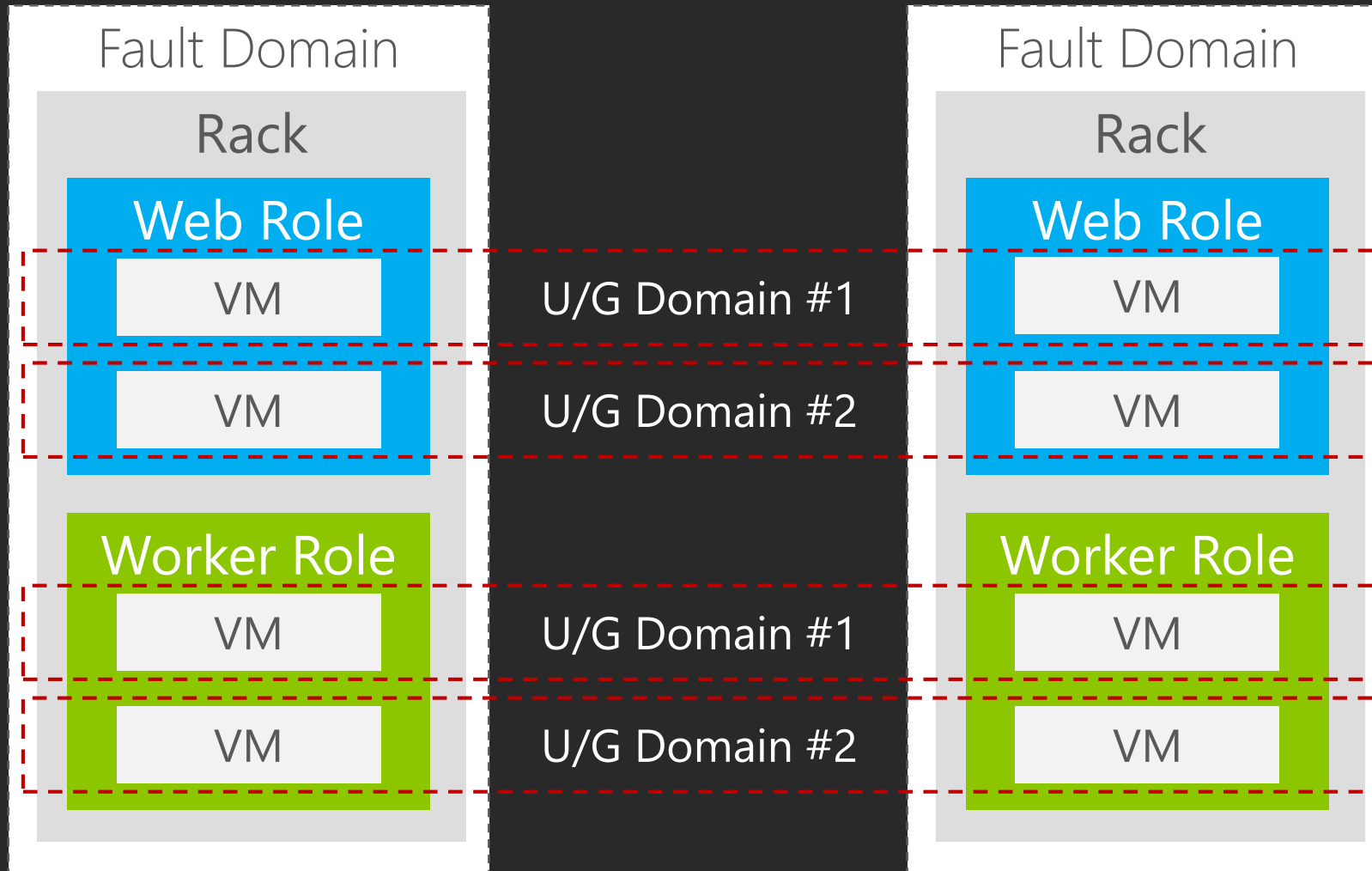
deployment/release strategies



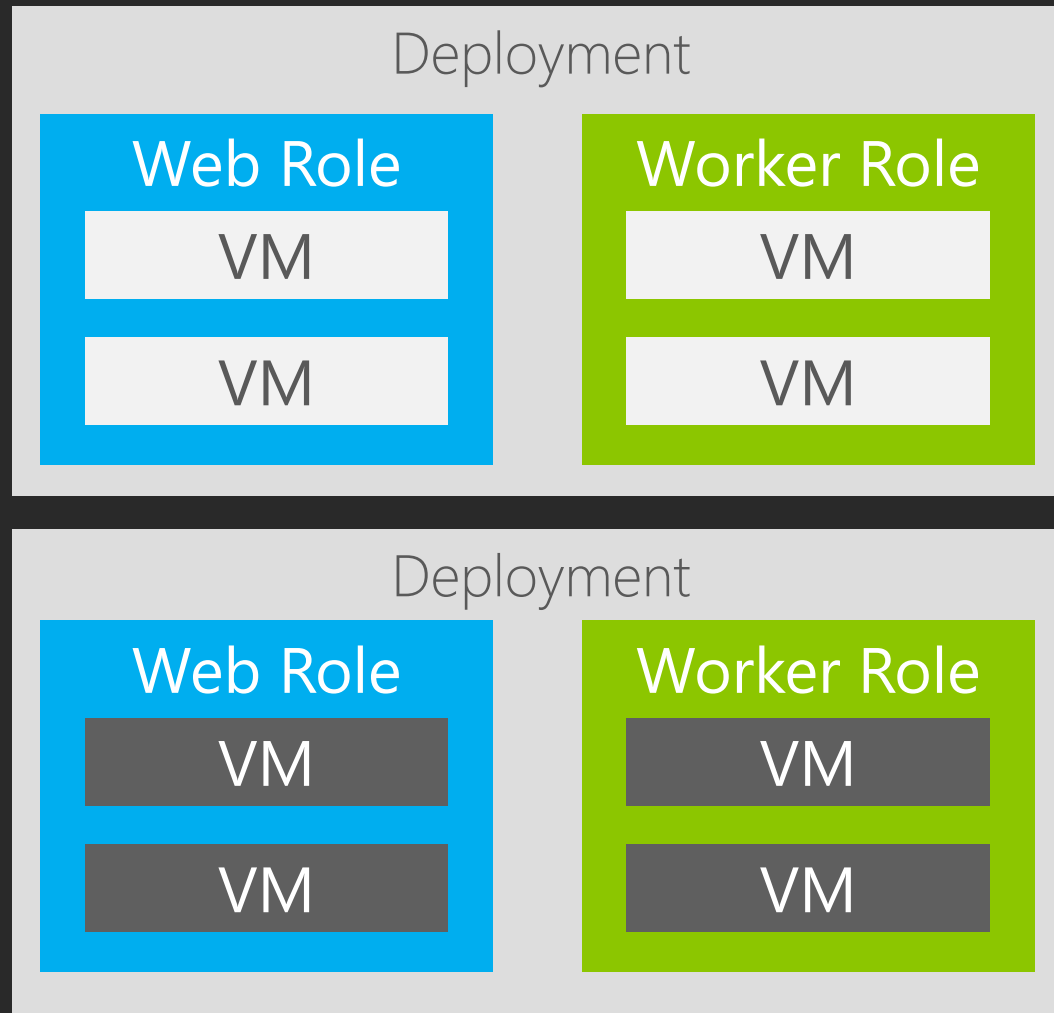
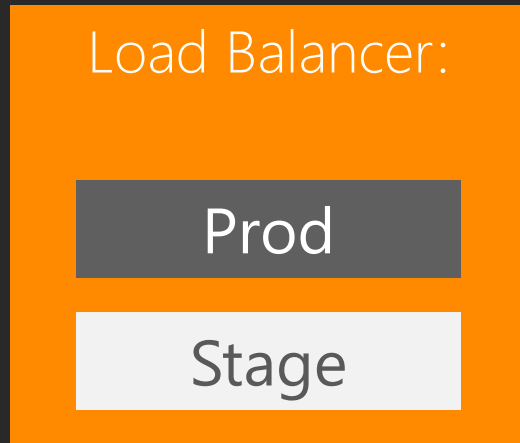
upgrades in Windows Azure



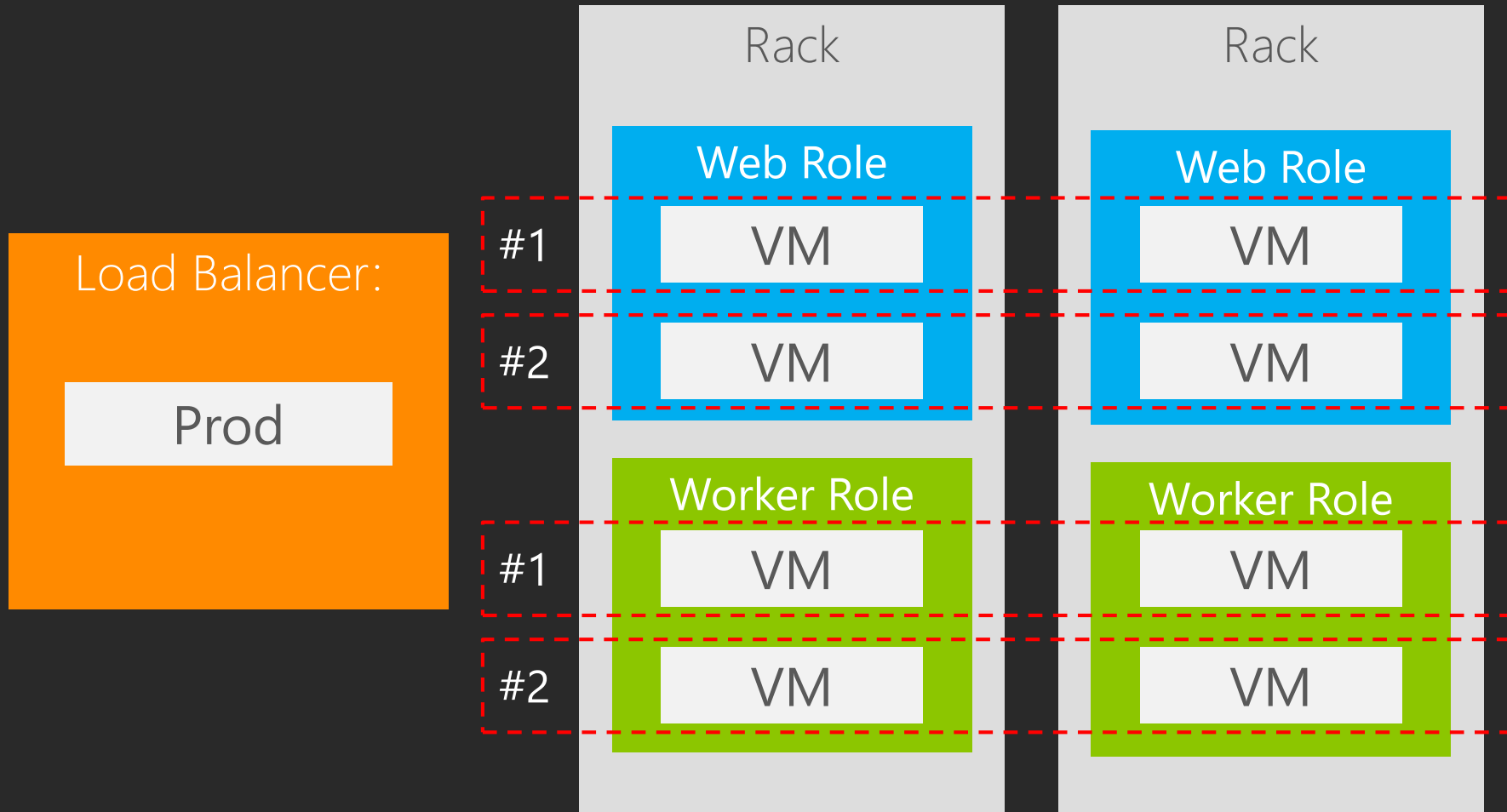
fault and upgrade domains



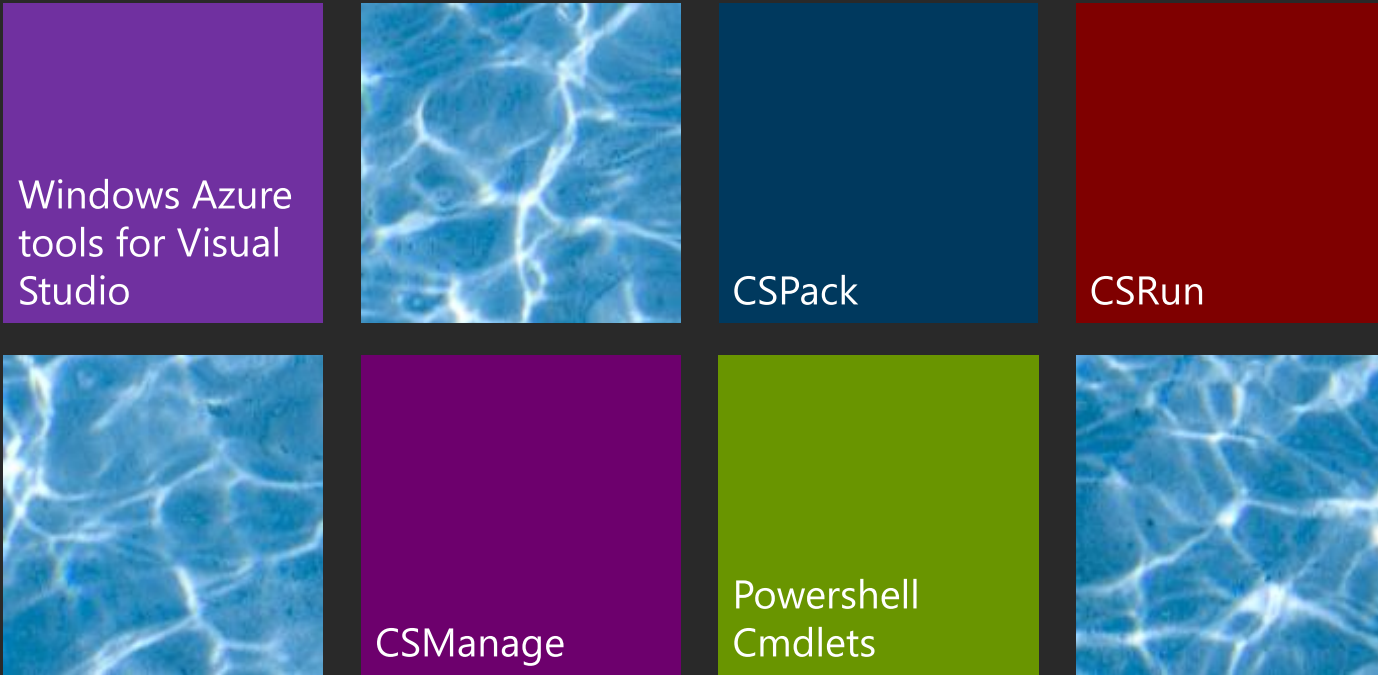
VIP swap

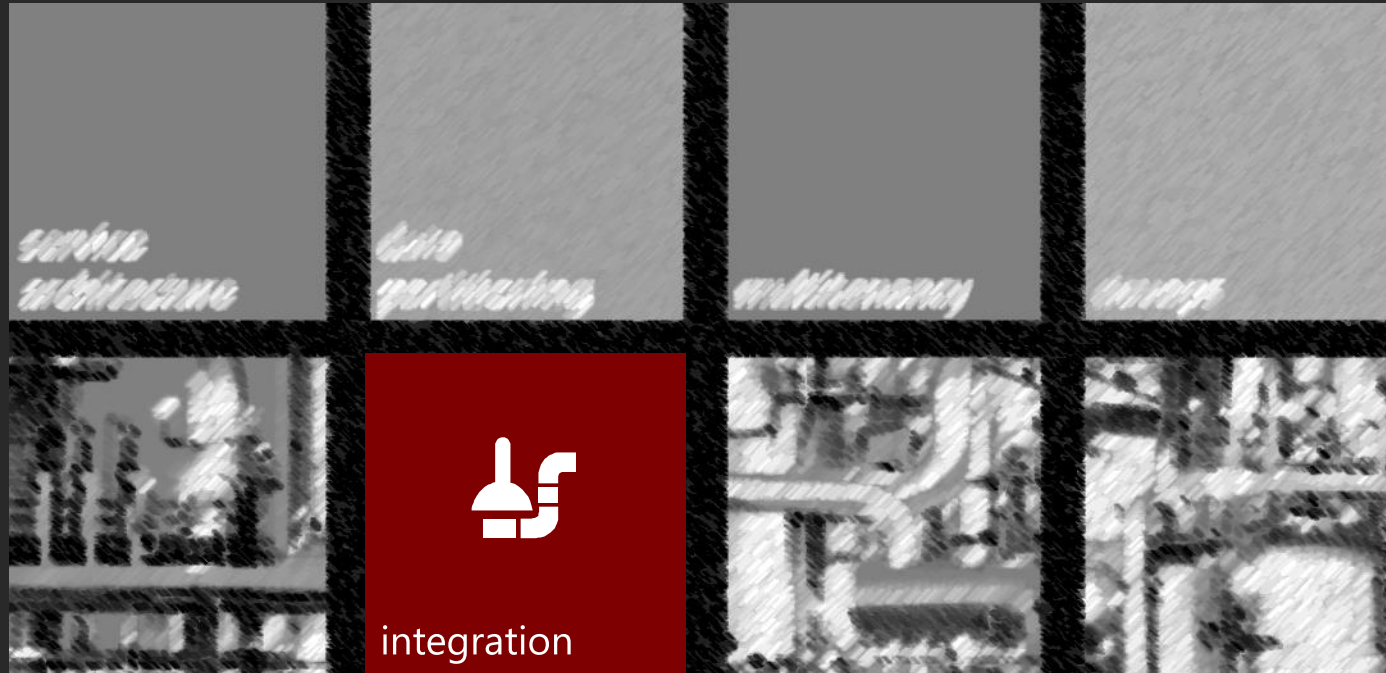


in place upgrade




Windows Azure deployment tools

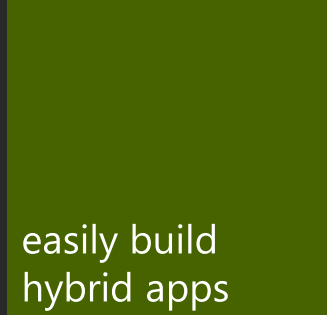




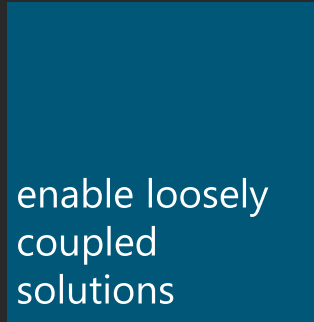
Windows Azure service bus



secure
messaging and
relay capabilities



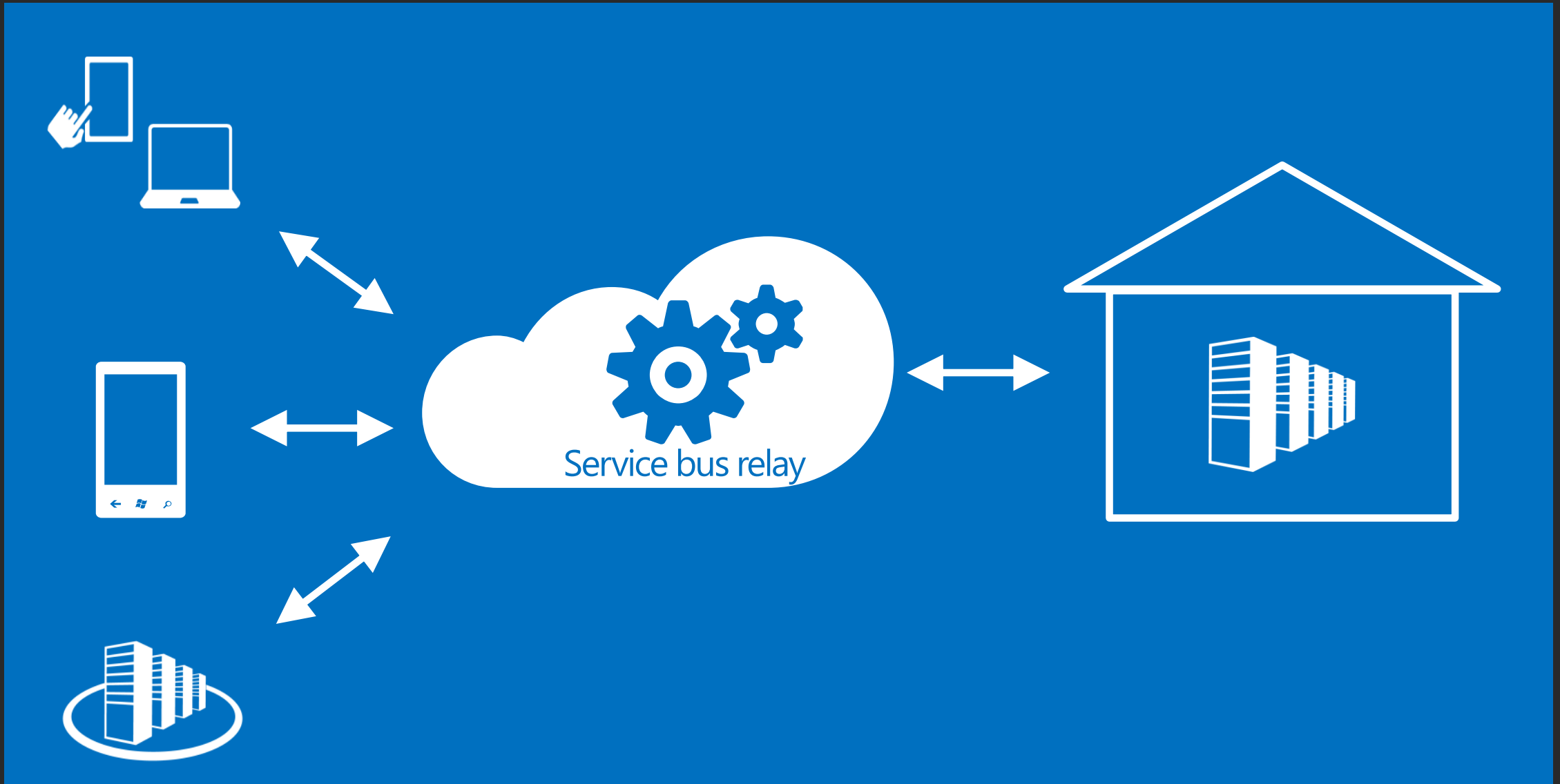
easily build
hybrid apps




enable loosely
coupled
solutions



accessing on-premise services





Demo:
Service Bus
Relay

conclusion



cloud style computing designs for scale out

scale out requires partitioning



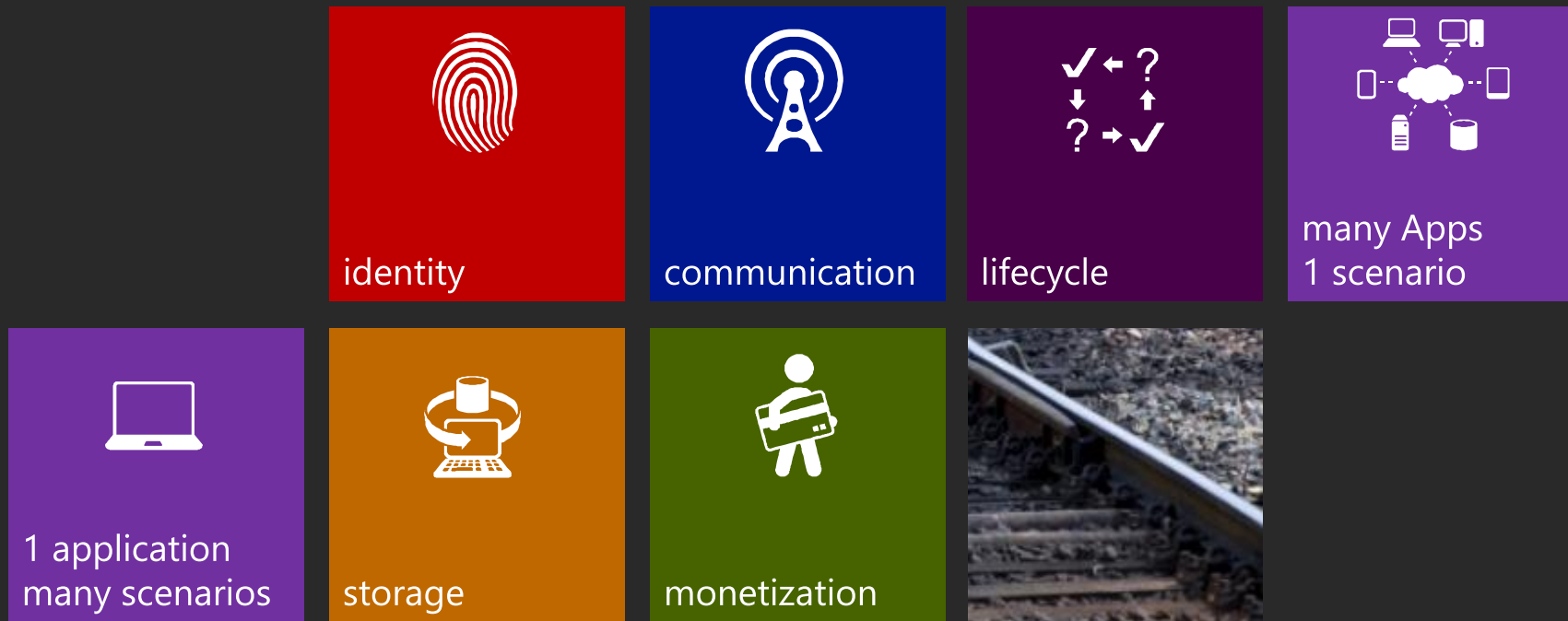
having redundant/duplicated data is ok

continuous delivery is a key asset



chapter V

from applications to apps



conclusion



from applications towards Apps

scenarios and tasks span multiple devices



cloud is a key enabler for connected devices

Windows Azure supports all devices

