SATURN 2015
Baltimore, MD — April 27-30, 2015

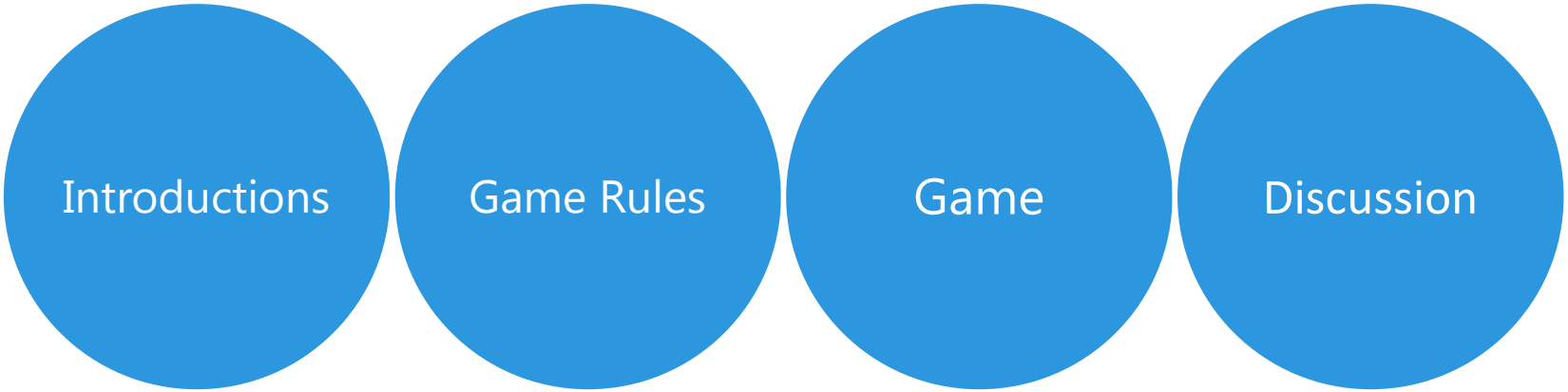
Smart Decisions

An Architecture Design Game

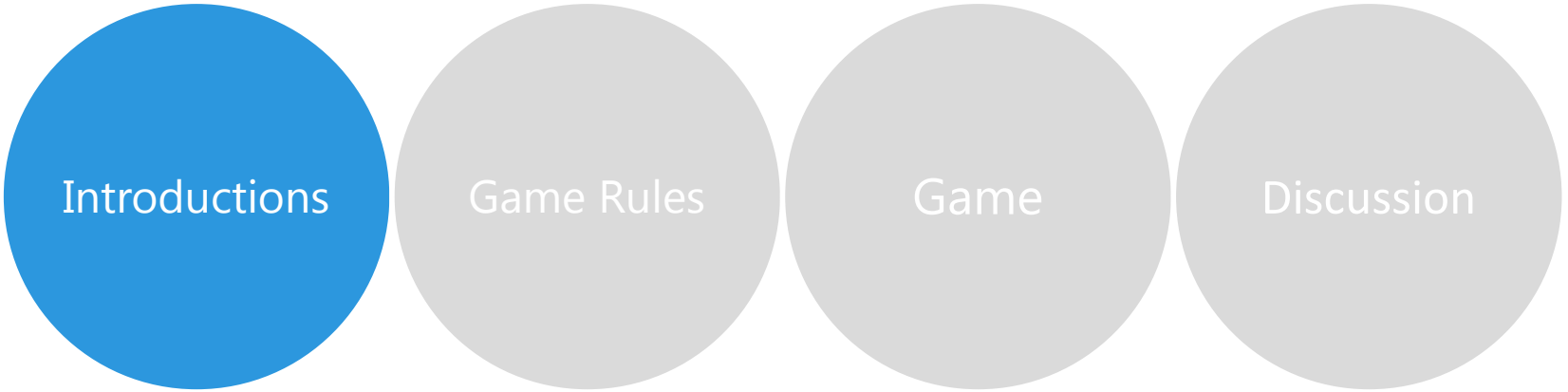
Humberto Cervantes, Serge Haziyev, Olha Hrytsay, Rick Kazman

April 2015

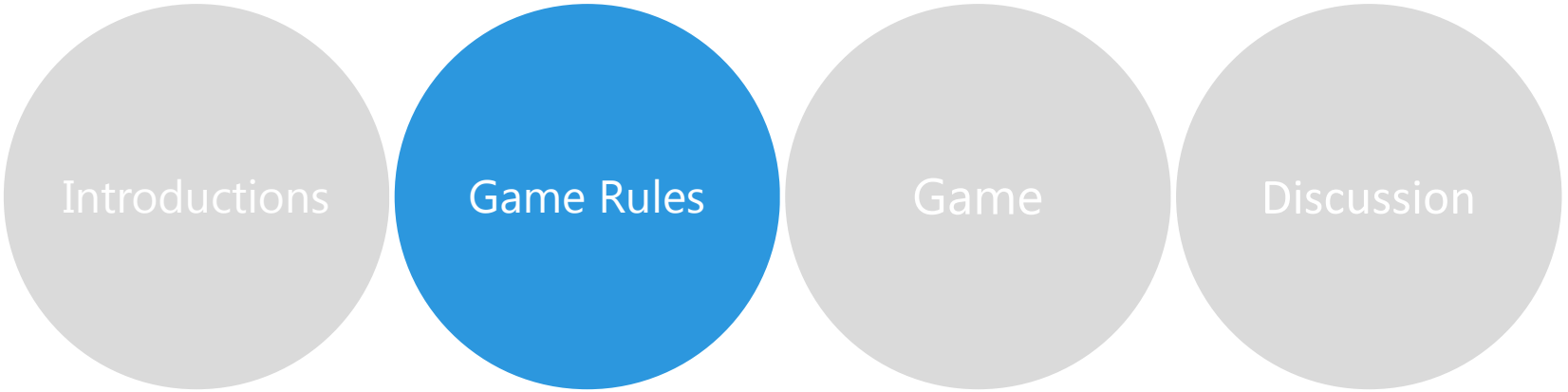
Agenda



Agenda



Agenda



Instructions

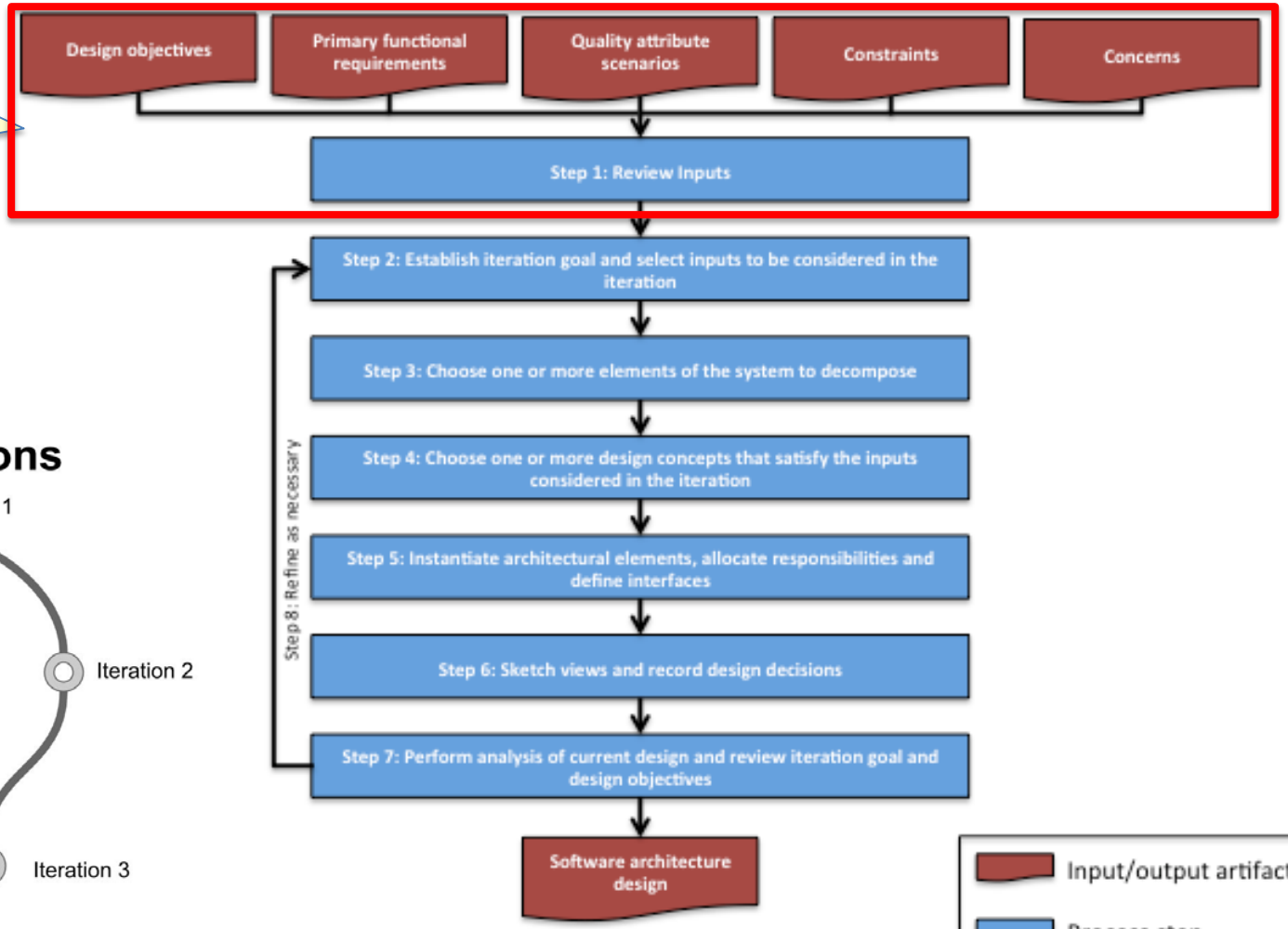
This game intends to illustrate the essentials of architecture design using an iterative method such as ADD.

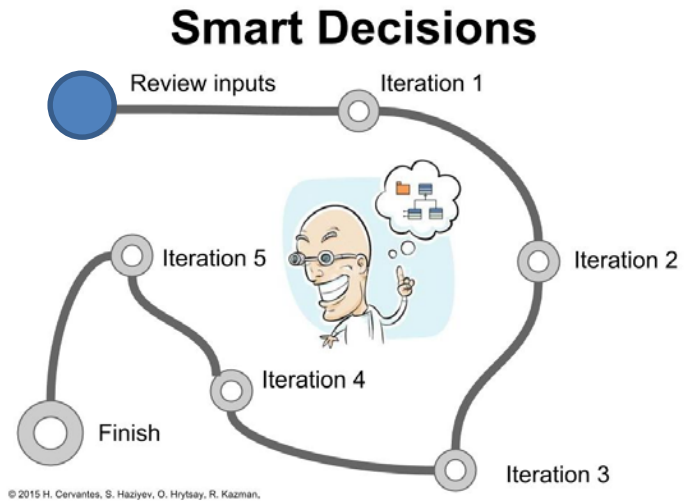
You will be competing against other software architects (or other teams) from rival companies, so you need to make *smart design decisions* or else your competitors will leave you behind!

Introduction

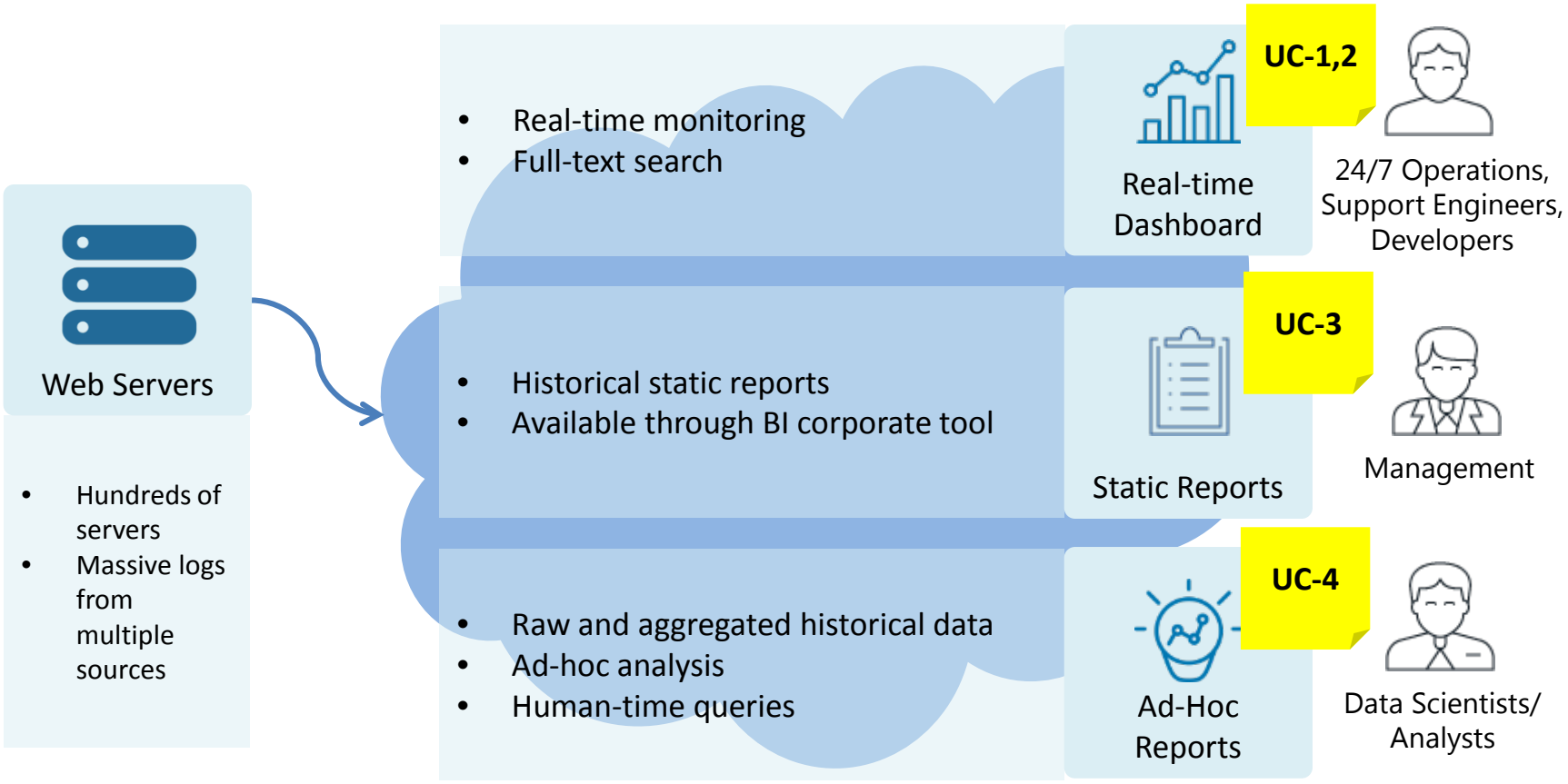
ADD Step 1: Review Inputs

Let's start by reviewing the inputs to the design process…

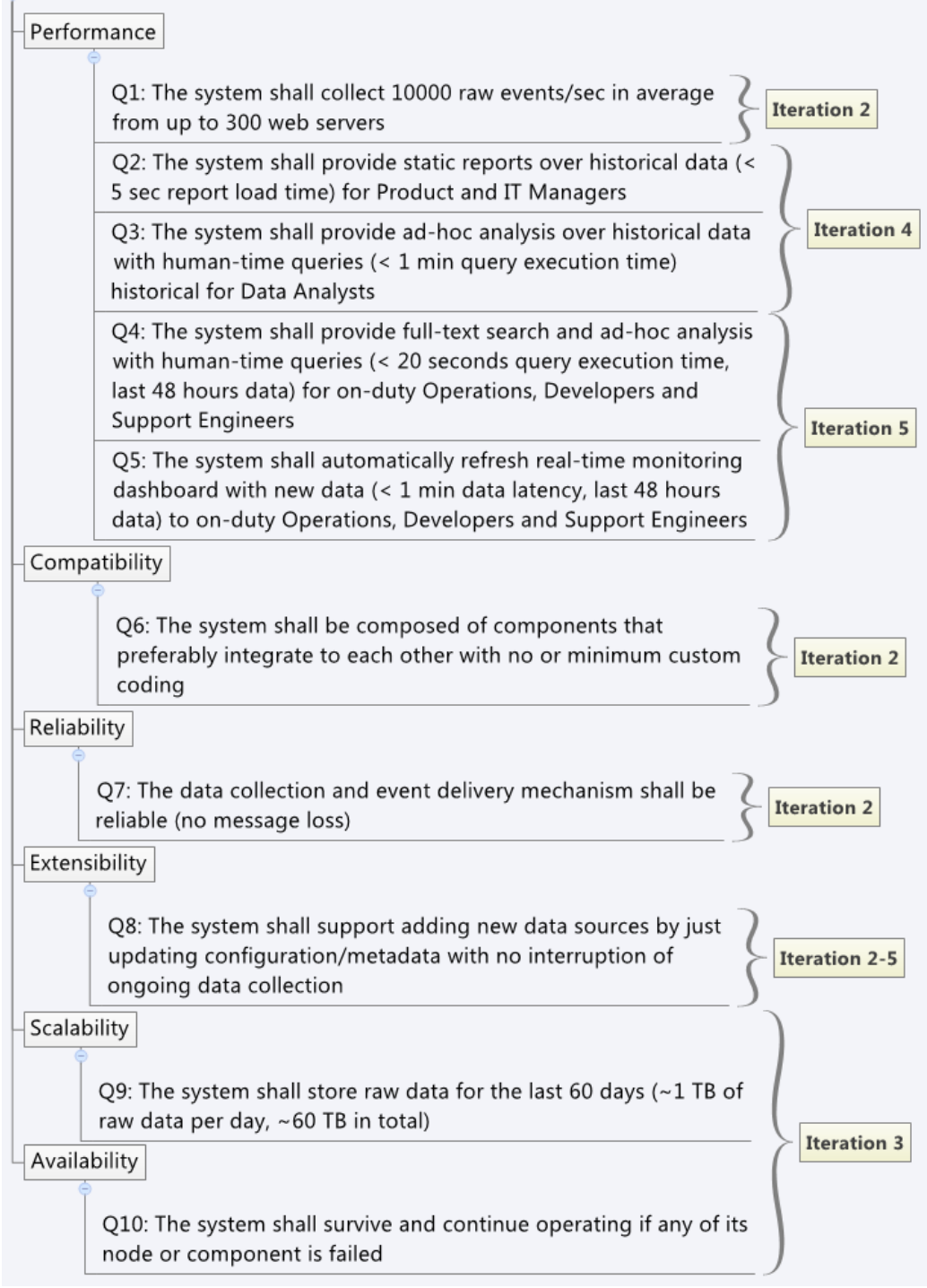




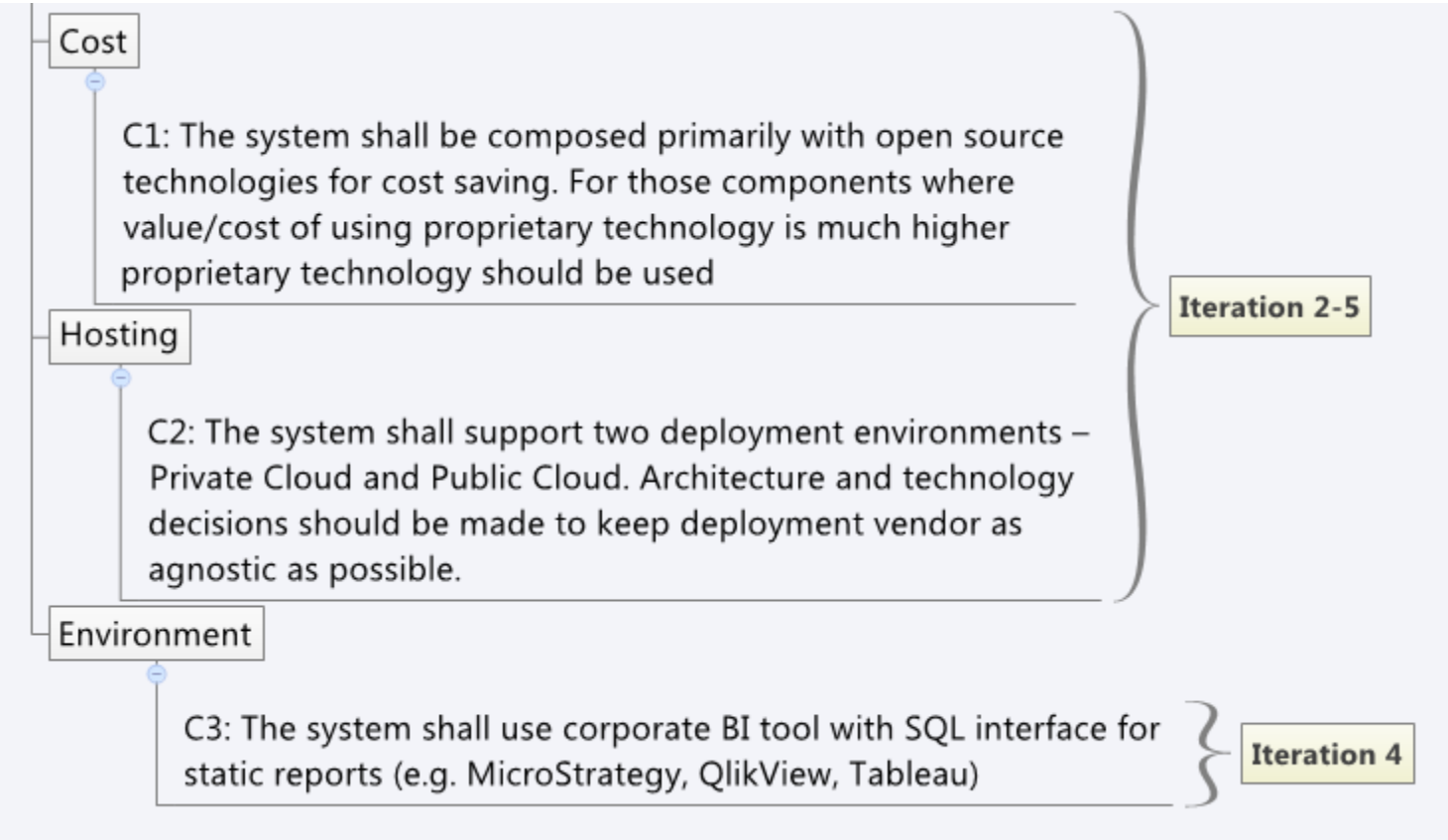
Functional drivers



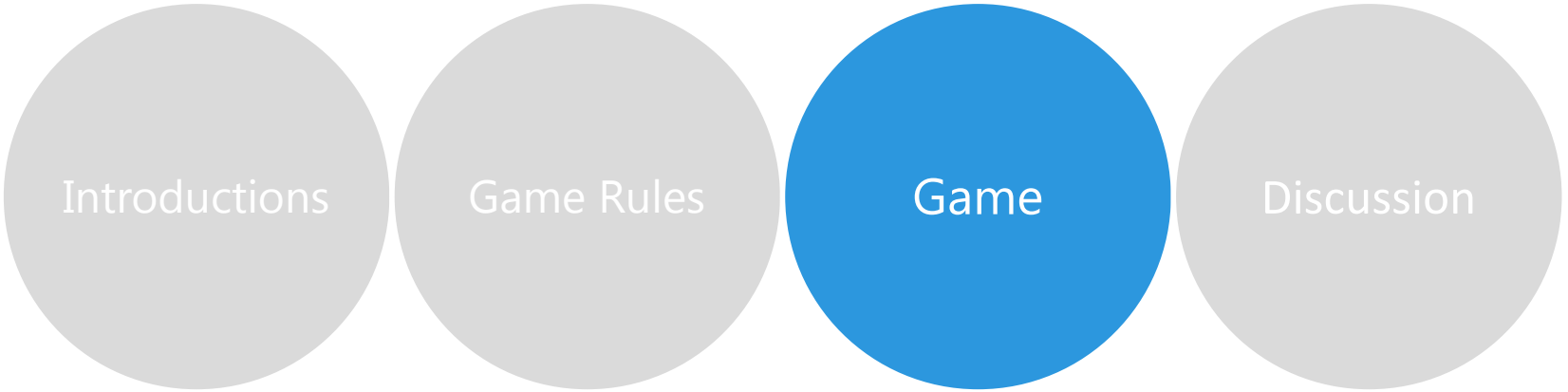
Quality attributes



Constraints



Agenda

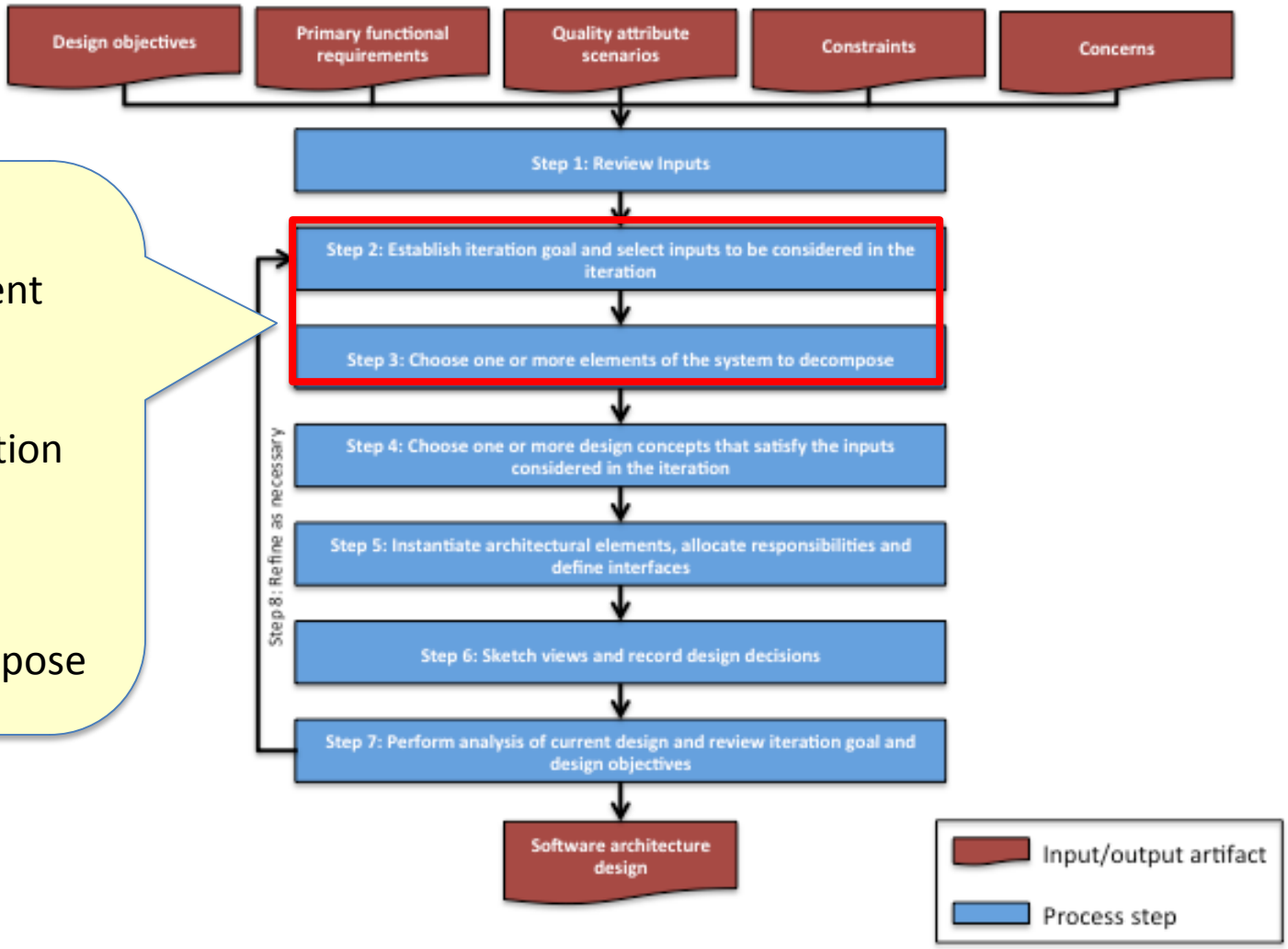


Game Rules

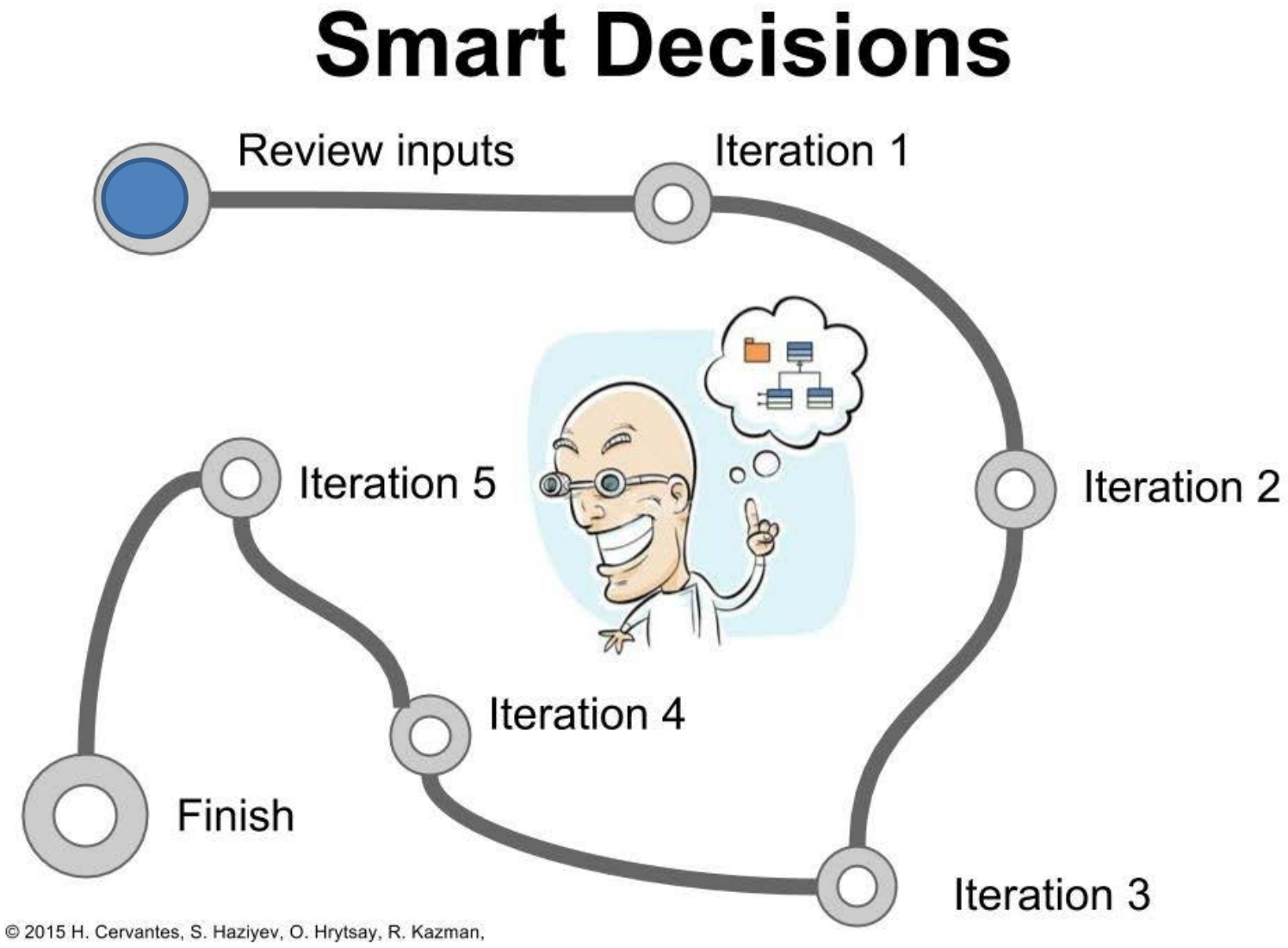
ADD Step 2: Review iteration goal and select inputs
ADD Step 3: Choose one or more elements of the system to decompose

The game is played in rounds which represent the iterations.

The goal for the iteration is provided:
- Drivers to be considered
- Element to decompose



Instructions



Iteration 1 goal: Logically structure the system

Drivers for the iteration:

- Ad-Hoc Analysis
- Real-time Analysis
- Unstructured data processing
- Scalability
- Cost Economy

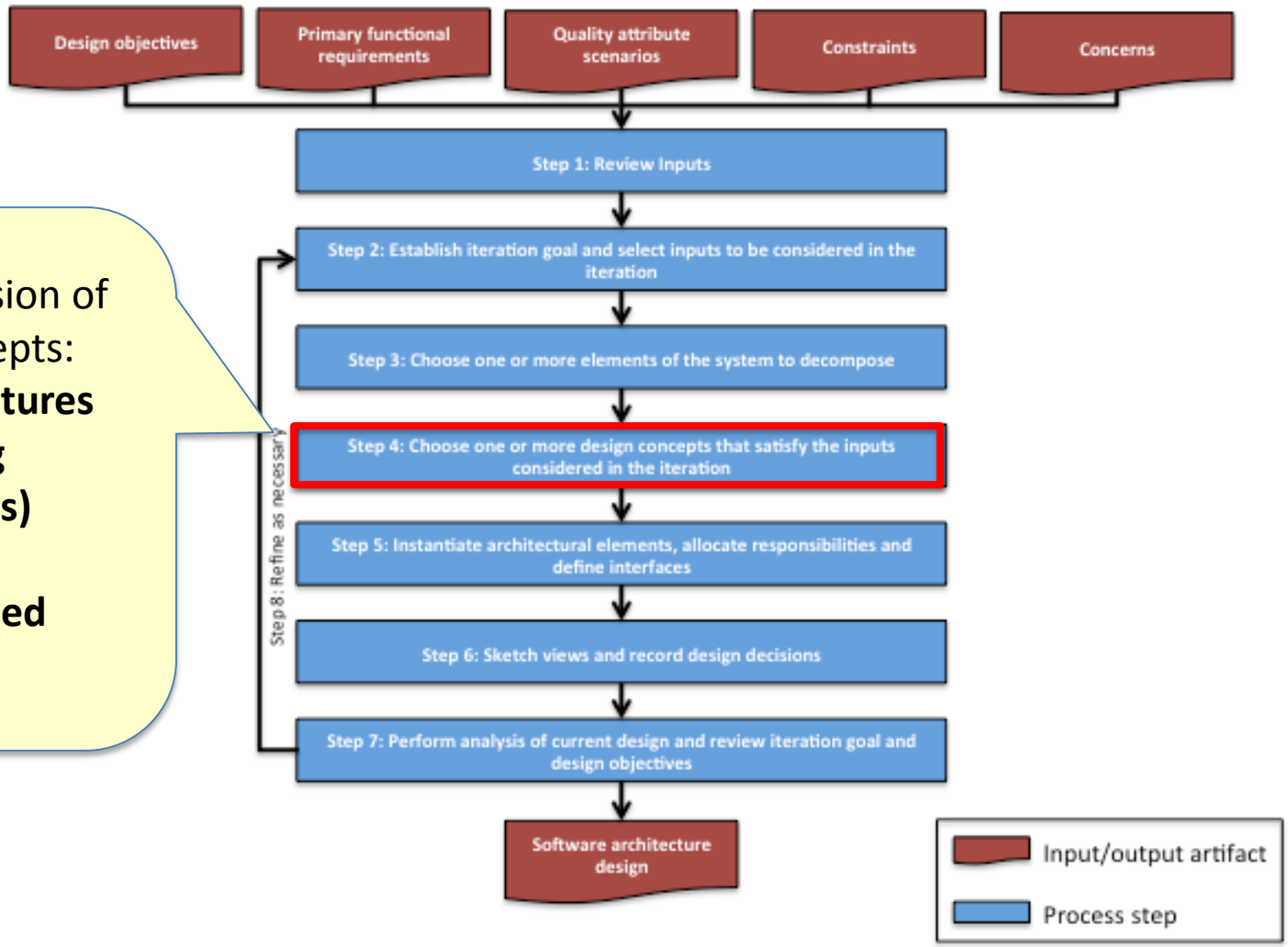
Element to decompose:

Game Rules

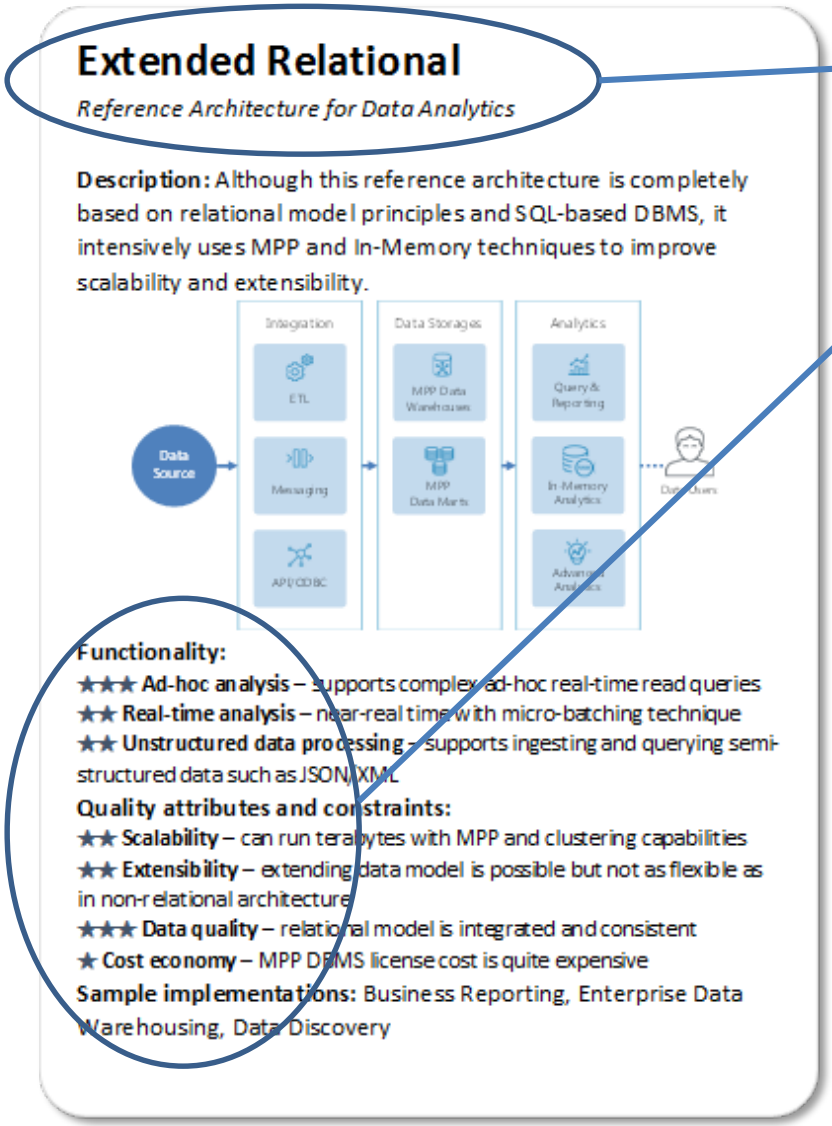
ADD Step 4: Choose one or more design concepts that satisfy the inputs considered in the iteration

Make the design decision of selecting design concepts:

- **Reference architectures**
- **Patterns (including technology families)**
- Tactics
- **Externally developed components**

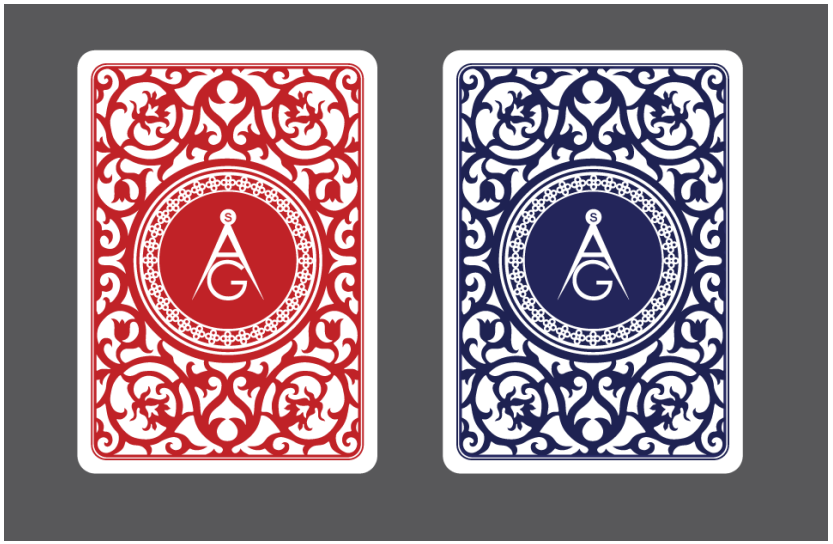


Game Rules: Design Concepts Cards



Name and type of design concept

Influence on drivers



Patterns **Technologies**

- Reference Architectures
- Families

Time to make your first smart decision!

Drivers for the iteration:

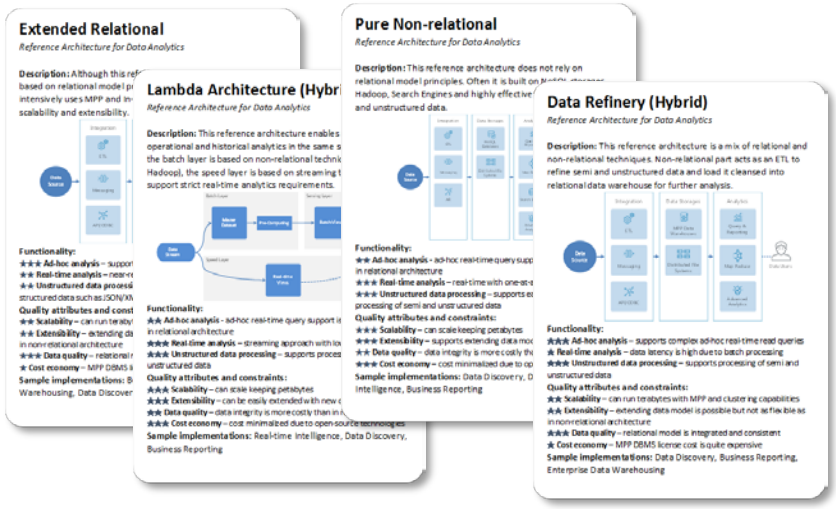
- Ad-Hoc Analysis
- Real-time Analysis
- Unstructured data processing
- Scalability
- Cost Economy

Element to decompose:

Big Data System

To Do:

Select 1 Reference Architecture Card



Possible alternatives:

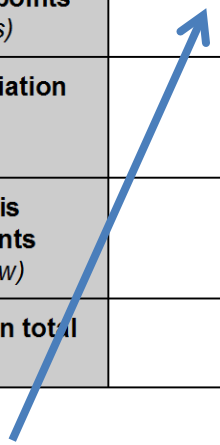
- Extended Relational
- Pure Non-Relational
- Data Refinery
- Lambda Architecture

Disqualified alternatives:

- Traditional Relational

Fill the scorecard

	Iteration #1	Iteration #2	Iteration #3	Iteration #4	Iteration #5	
(a) Design Decisions *(Names of selected design concept(s))*						
(b) Driver selection points *(from cards)*						
(c) Instantiation points *(from dice)*						
(d) Analysis bonus points *(from review)*						**Final score:**
(e) Iteration total *(b + c + d)*						



Fill (b) by adding the points for the drivers considered for the iteration, in this case:

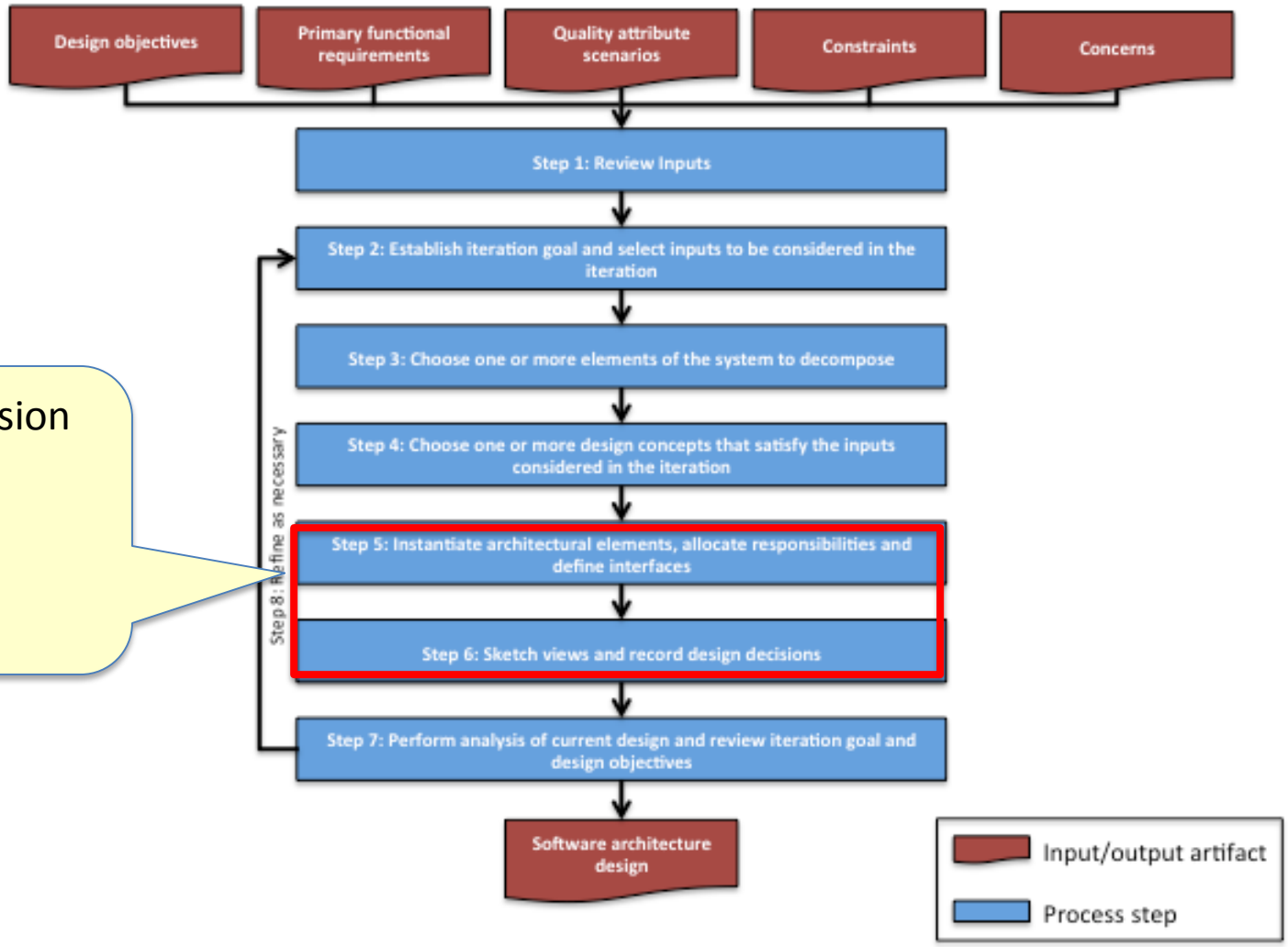
- Ad-Hoc Analysis
- Real-time Analysis
- Unstructured data processing
- Scalability
- Cost Economy

 **= 1 Point**

Introduction

ADD Step 5: Instantiate elements, allocate responsibilities and define interfaces.
ADD Step 6: Sketch views and record design decisions

Record the design decision and throw two dice to simulate how well you instantiate your design concept



Fill the scorecard

	Iteration #1	Iteration #2	Iteration #3	Iteration #4	Iteration #5	
(a) Design Decisions *(Names of selected design concept(s))*						
(b) Driver selection points *(from cards)*						
(c) Instantiation points *(from dice)*						
(d) Analysis bonus points *(from review)*						**Final score:**
(e) Iteration total *(b + c + d)*						

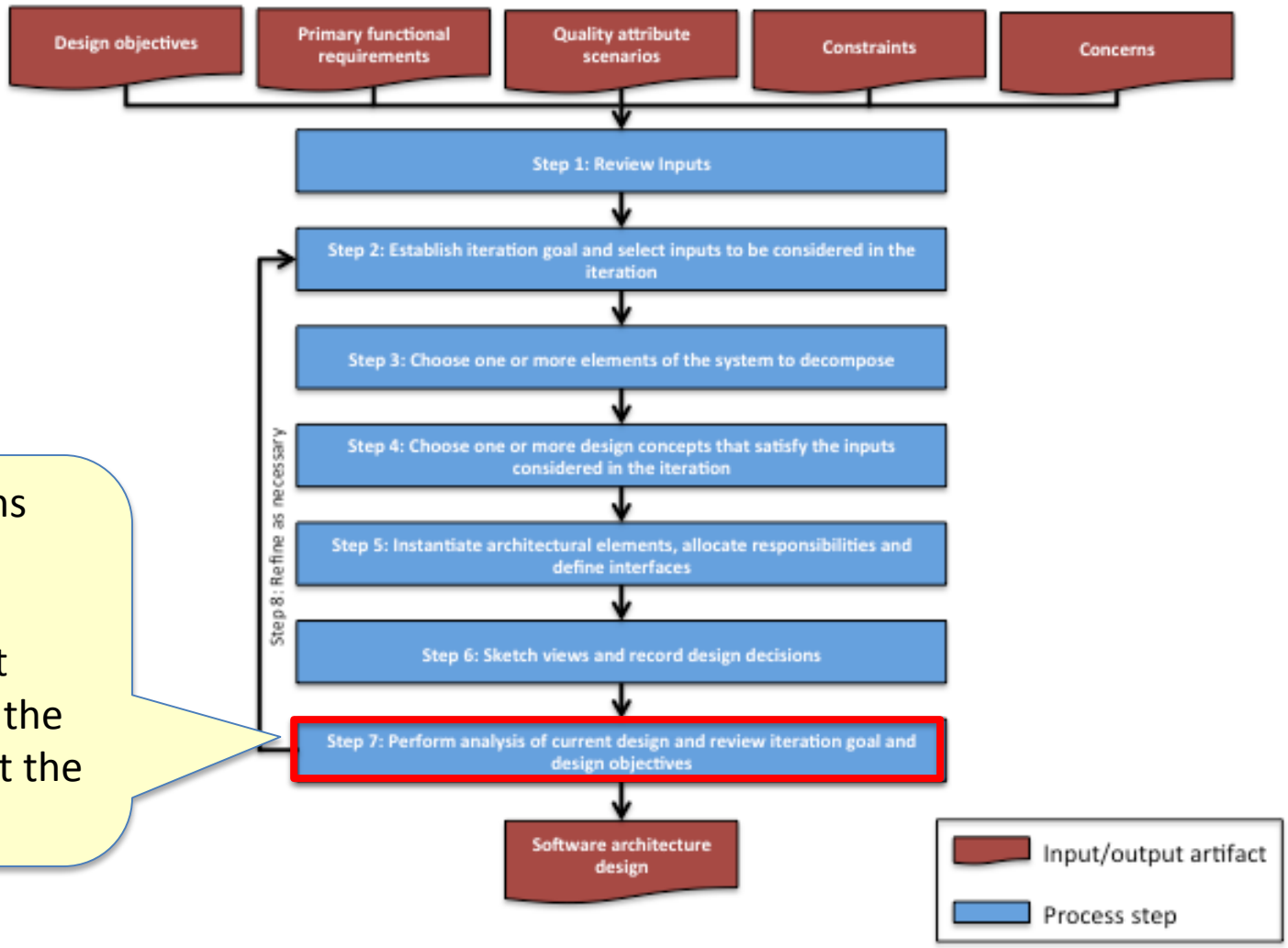
Record design decisions in (a)

Roll the dice and add or subtract points according to the following table, fill (c).

Dice result[1]	Points
2 - 3	-2
4 - 9	0
10 - 12	+2

Introduction

ADD Steps



Review design decisions and score iteration.

We will review the first iteration together, but the rest will be reviewed at the end.

Iteration 1: Scoring

Score Ad-Hoc Analysis, Real-time Analysis, Unstructured data processing, Scalability, Cost Economy

Design decision	Driver points	Bonus points	Comments
Extended Relational	3+2+2+2+1=**10**	**-4**	This reference architecture is less appropriate for this solution mostly because of cost and real-time analysis limitation
Pure Non-Relational	2+2.5+3+3+3=**13.5**		This reference architecture is closer to the goal than the others except Lambda Architecture
Lambda Architecture (Hybrid)	2.5+3+3+3+3=**14.5**	**+2**	This is the most appropriate reference architecture for this solution! From the provided reference architectures Lambda Architecture promises the largest number of benefits, such as access to real-time and historical data at the same time.
Data Refinery (Hybrid)	3+1+3+2+1=**10**	**-4**	This reference architecture is less appropriate for this solution mostly because of cost and real-time analysis limitation

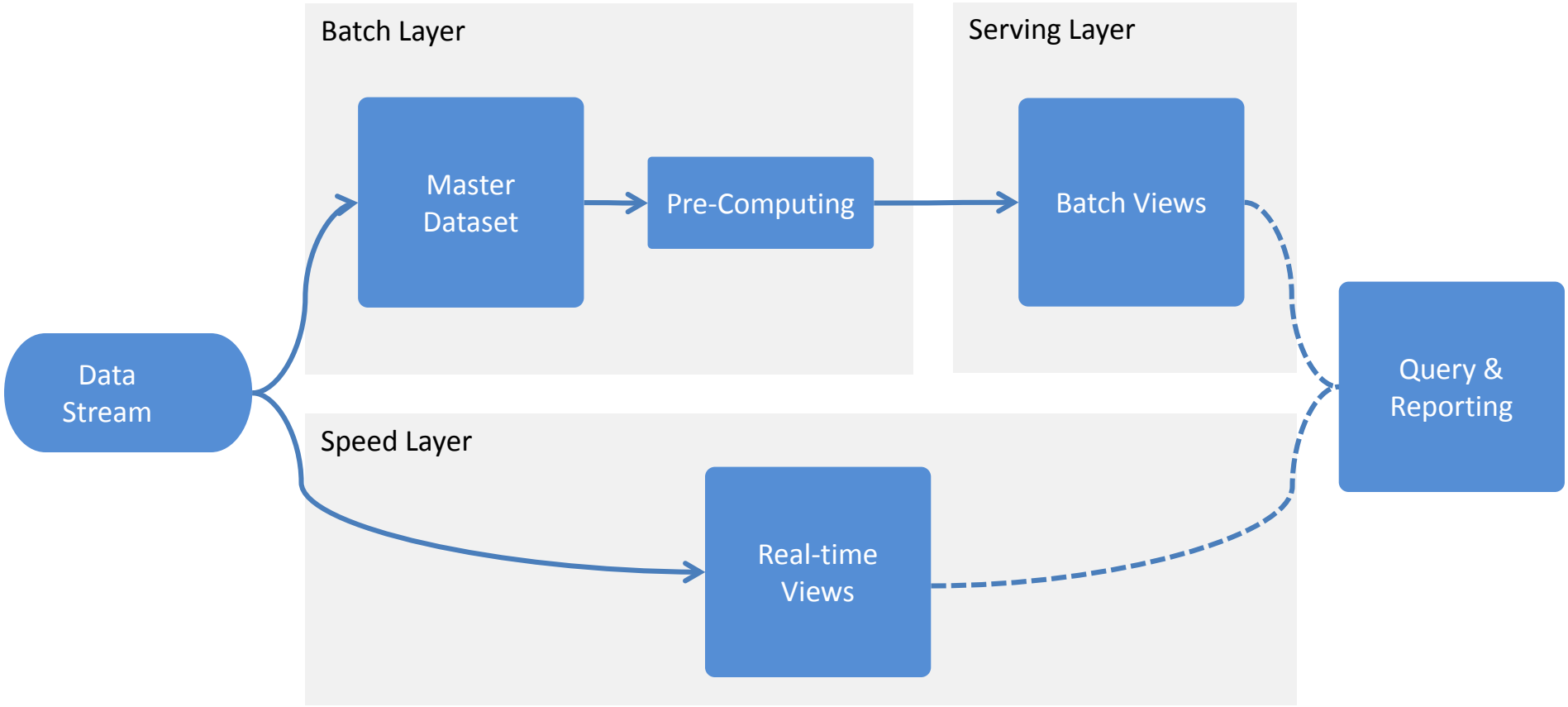
Fill the scorecard

	Iteration #1	Iteration #2	Iteration #3	Iteration #4	Iteration #5	
(a) Design Decisions *(Names of selected design concept(s))*						
(b) Driver selection points *(from cards)*						
(c) Instantiation points *(from dice)*						
(d) Analysis bonus points *(from review)*						**Final score:**
(e) Iteration total *(b + c + d)*						

Add bonus points, if any and fill (d)

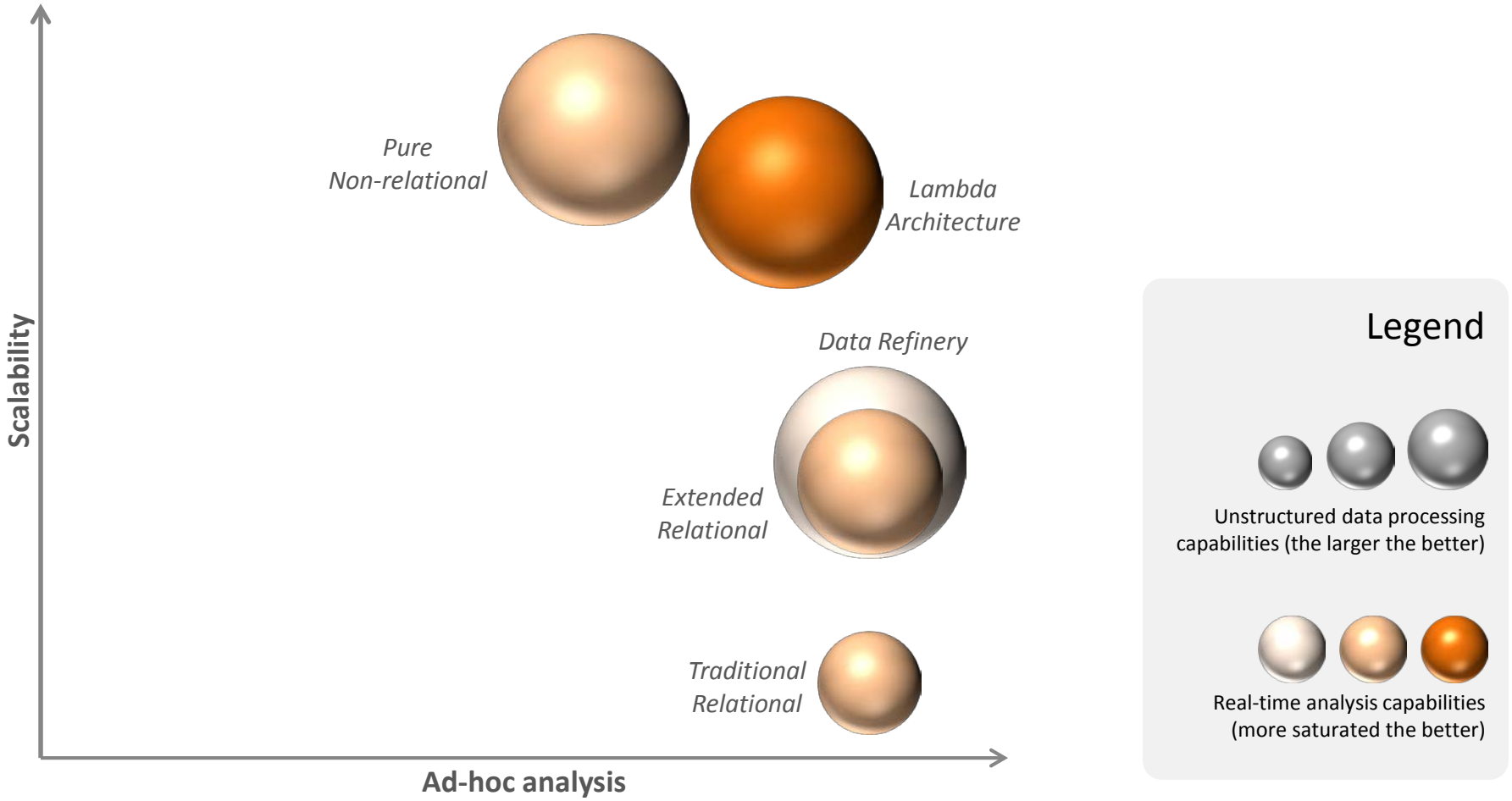
Sum the points and calculate the total for the iteration in (e)

Lambda Architecture Logical Structure

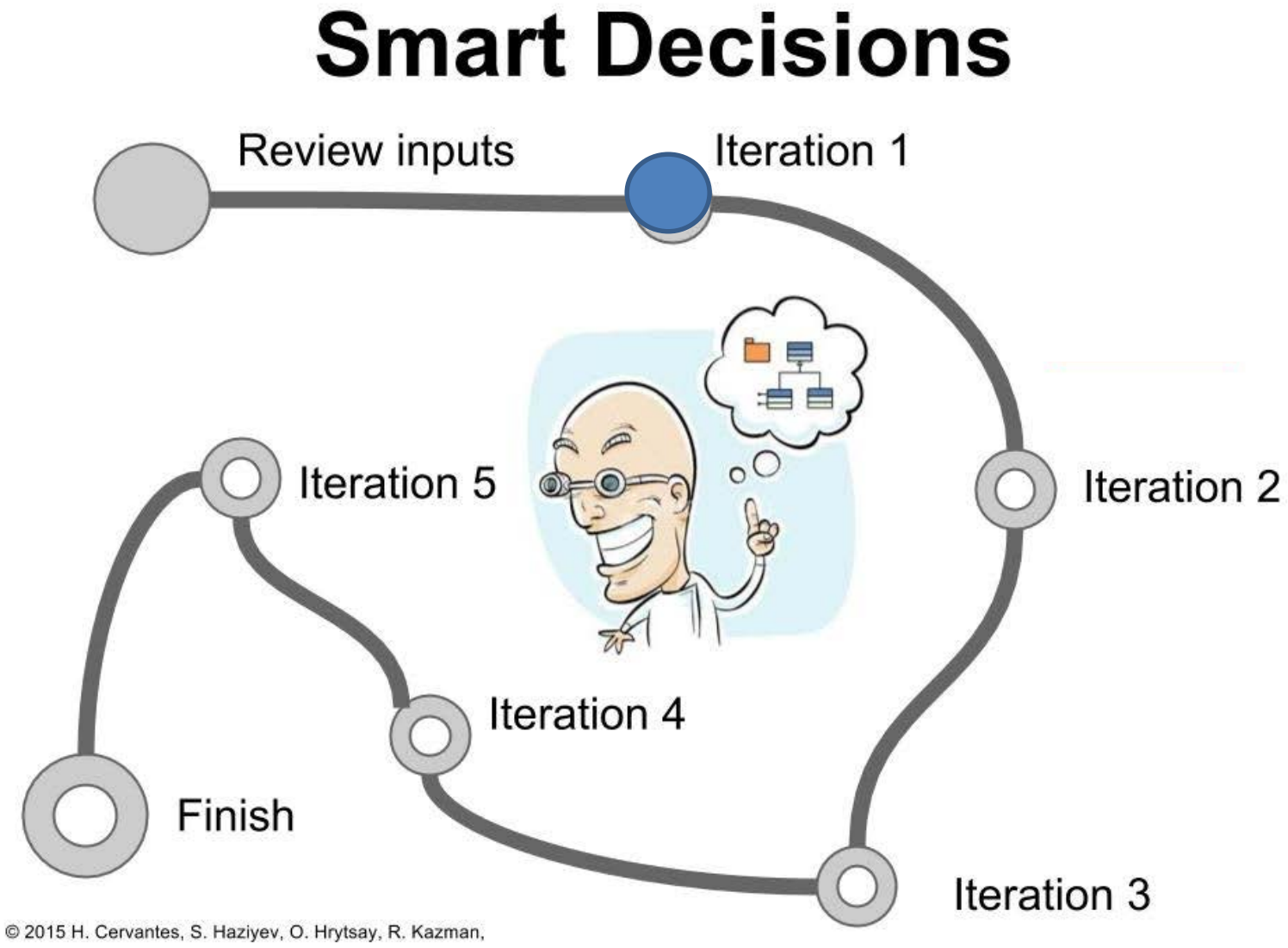


Source: http://lambda-architecture.net/

Big Data Analytics Reference Architectures Trade-off



Instructions

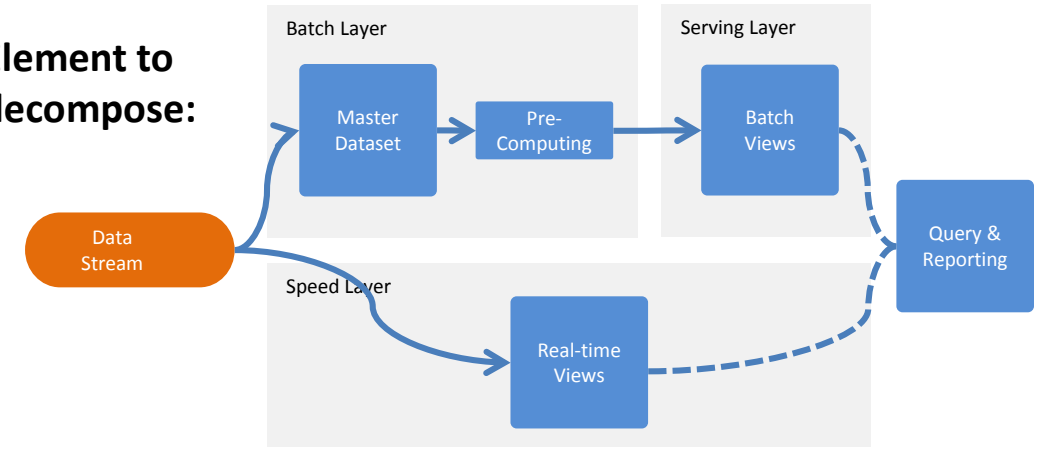


Iteration 2: Design Data Stream Element

Drivers for the iteration:

- Performance (for Family and Technology)
- Compatibility (for Family)
- Reliability (for Technology)

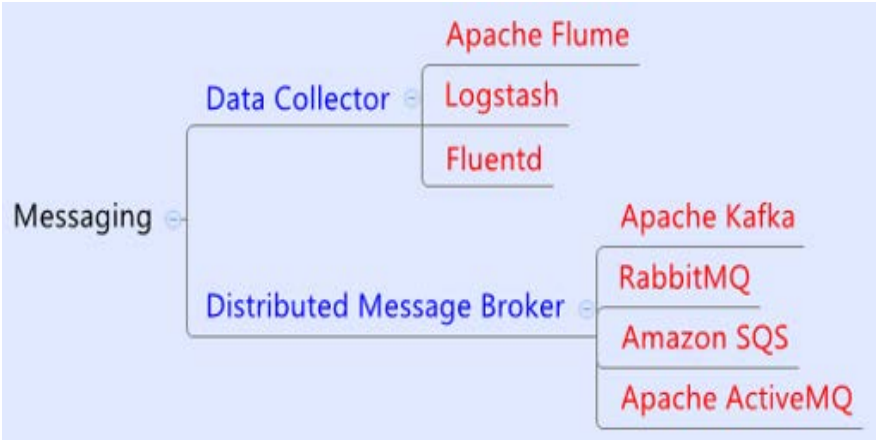
Element to decompose:



To Do:

Select 1 Family card and 1 Technology card

Possible alternatives:



Disqualified alternatives:

- ETL Engine (lack of real-time data stream support and no need for complex data transformations)

Tip:

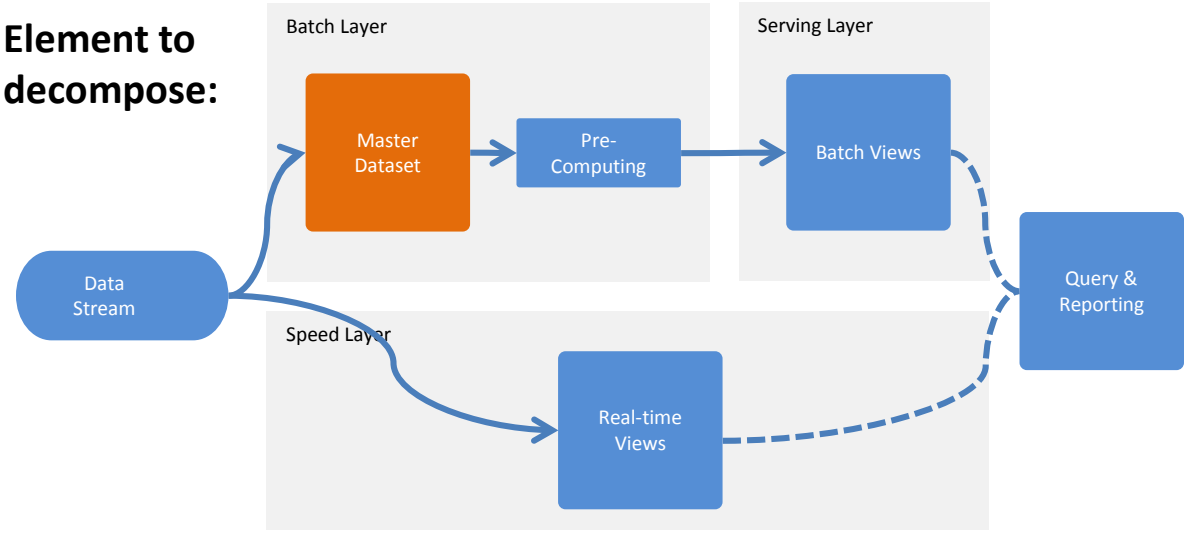
- Look for an option that can be deployed on-Premise and on-Cloud

Iteration 3: Design Batch Layer

Drivers for the iteration:

- Scalability
- Availability

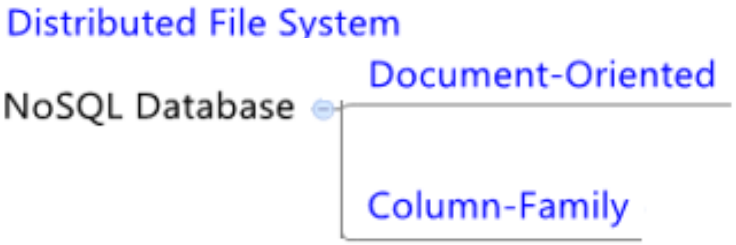
Element to decompose:



To Do:

Select 1 Family card

Possible alternatives:



Disqualified alternatives:

- NoSQL Database/Key-Value
- NoSQL Database/Graph-Oriented
- Analytic RDBMS
- Distributed Search Engine

Tip:

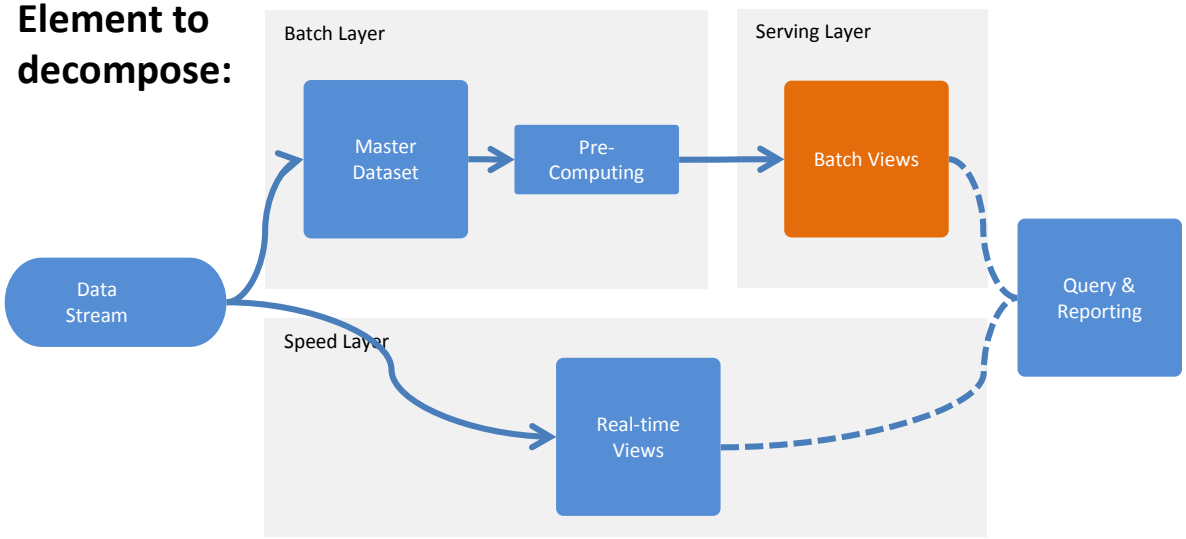
- Look for an option with better extensibility (easy storing of new data formats)

Iteration 4: Design Serving Layer

Drivers for the iteration:

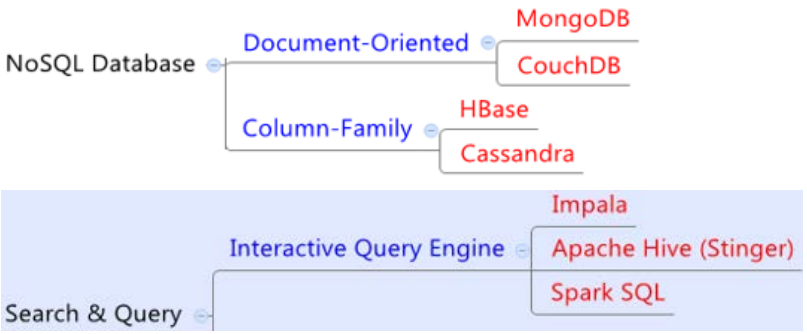
- Ad-hoc Analysis (for Family)
- Performance (for Family and Technology)

Element to decompose:



To Do: Select 1 Family and 1 Technology card

Possible alternatives:



Disqualified alternatives:

- NoSQL Database/Key-Value
- NoSQL Database/Graph-Oriented
- Analytic RDBMS
- Distributed Search Engine

Tip:

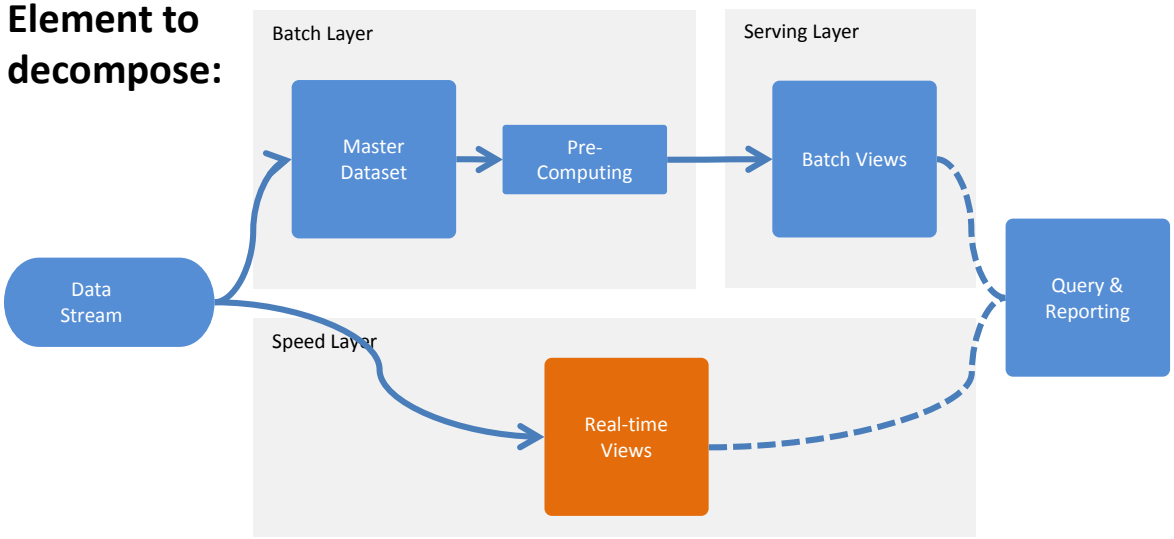
- Look for an option that provides ad-hoc analysis and still good performance for static reports

Iteration 5: Design Speed Layer

Drivers for the iteration:

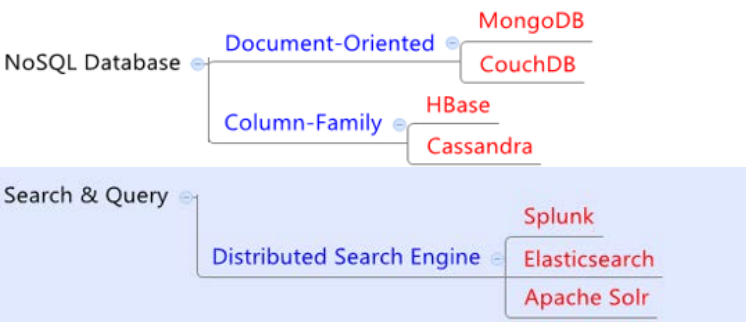
- Ad-hoc Analysis (for the family)
- Real-time Analysis (for the technology)

Element to decompose:



To Do: Select 1 Family and 1 Technology card

Possible alternatives:



Disqualified alternatives:

- NoSQL Database/Key-Value
- NoSQL Database/Graph-Oriented
- Analytic RDBMS

Tip:

- Look for an option that provides full-text search capabilities and extensibility (new data formats and dashboard views)

Iteration 2: Design decisions analysis and scoring

Family card: score Performance and Compatibility

Design decision	Driver points	Bonus points	Comments
Data Collector	2+3=**5**	**+2**	Additional bonus is added for extensibility
Distributed Message Broker	3+1=**4**		

Technology card: score Performance and Reliability

Design decision	Driver points	Bonus points	Comments
Apache Flume	2+2=**4**		
Logstash	2+2=**4**		
Fluentd	2+3=**5**		
RabbitMQ	2+2=**4**		
Apache Kafka	3+2=**5**	**+2**	Additional bonus for easier deployment and configuration comparing with other alternatives
Amazon SQS	0		Disqualified due to deployment constraint (support On-premise and Cloud)
Apache ActiveMQ	2+2=**4**		

Iteration 3: Design decisions analysis and scoring

Family card: score Scalability and Availability

Design decision	Driver points	Bonus points	Comments
NoSQL Database/Column-Family	3+3=**6**	**-1**	Column families must be defined up front and require modification when log format is changed – extensibility disadvantage
NoSQL Database/Document-Oriented	3+3=**6**		
Distributed File System	3+3=**6**	**+2**	Bonus for extensibility (log format changes do not require any changes in DFS cluster) and easier deployability/maintainability compared with NoSQL databases

Note: If you selected FluentD during the previous iteration and DFS at this iteration you receive **-1 performance bonus** (FluentD uses WebHDFS which pays a little performance cost due to HTTP)

Iteration 4: Design decisions analysis and scoring

Family card: score Ad-hoc Analysis, Performance

Design decision	Driver points	Bonus points	Comments
Interactive Query Engine	3+2=**5**	**+2**	Extensibility bonus because this approach does not require complex tuning of schema for introducing new reports and data types
NoSQL Database/Column-Family (+ SQL connector)	1+3=**4**		
NoSQL Database/Document-Oriented (+ SQL connector)	1.5+3=**4.5**		

Technology card: score Performance

Design decision	Driver points	Bonus points	Comments
Impala	**3**		
Apache Hive	**1.5**		
Spark SQL	**3**		
Apache Cassandra	**3**		
Apache HBase	**2.5**		
MongoDB	**2**		
Apache CouchDB	**1.5**		

Iteration 5: Scoring

Family card: score Ad-hoc Analysis

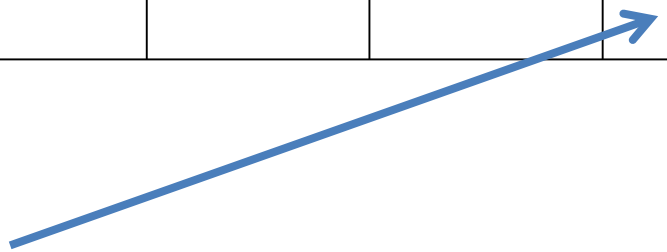
Design decision	Driver points	Bonus points	Comments
Distributed Search Engine	**2**	**+2**	Full-text search is out of the box + bonus for extensibility (adding new log formats and report views requires minimum changes in search engine)
NoSQL Database/Column-Family	**1**		
NoSQL Database/Document-Oriented	**1.5**		

Technology card: score Real-time Analysis

Design decision	Driver points	Bonus points	Comments
Elasticsearch	**2.5**	**+2**	Elasticsearch easily integrates with Kibana – an open source interactive dashboard
Apache Solr	**2.5**		
Splunk (Indexer)	**2.5**	**-2, +2**	-2 penalty for cost and +2 bonus (Splunk offers end-to-end solution including powerful visualization tool)
Apache Cassandra	**3**		
Apache HBase	**3**		
MongoDB	**3**		
Apache CouchDB	**3**		

Fill the scorecard

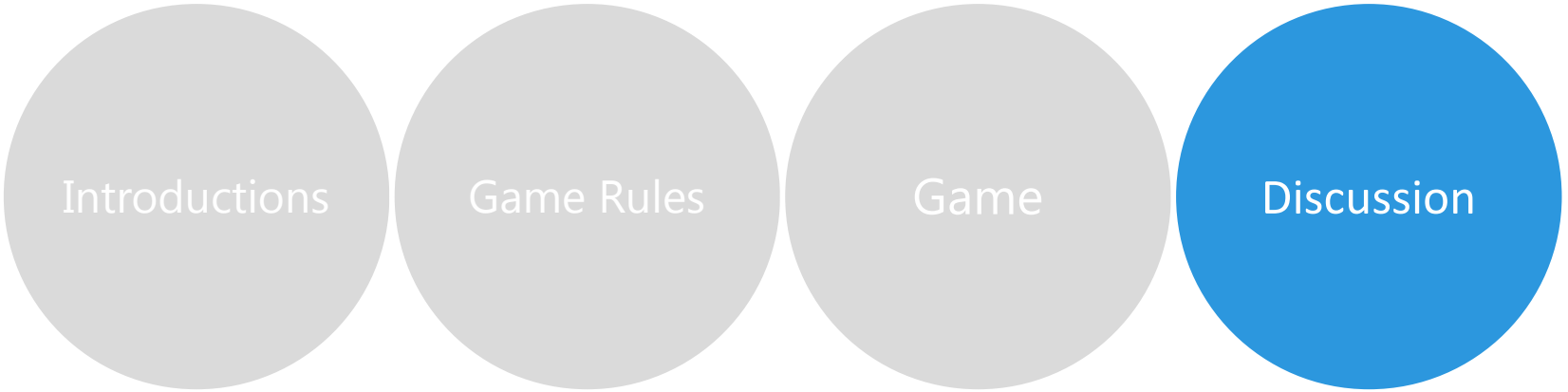
	Iteration #1	Iteration #2	Iteration #3	Iteration #4	Iteration #5	
(a) Design Decisions *(Names of selected design concept(s))*						
(b) Driver selection points *(from cards)*						
(c) Instantiation points *(from dice)*						
(d) Analysis bonus points *(from review)*						**Final score:**
(e) Iteration total *(b + c + d)*						



Calculate the final score

- Add 2 to the player who finished first
- Add 1 to the player who finished second

Agenda




Thank
You
Merci
Gracias
Danke
Spacibo
KamSahHamnida
Arigatô
Hvala
TerimaKasih
Mèsi
Xièxie
Takk
aDank
BarakAllahuFiik
Grazie
Chokrane
Kiitos
Dziakuju
GratiasAgimus
GratiasAgo
Motashakkeram
Tak
Dankie
Sagolun
KopKhunKha
Gracie
CámOn
Multumesc
Saha
Salamat
Obrigado
Bayarlalaa
Nirringrazzjak
NajjsTuke
Efharisto
Faleminderit
Gràcies
Grandmercé
Taiku
Dakujem
Mercé
Matondo
Blagodaram
ThintKo
Buznyg
TapadhLeat
Köszönöm
Murakoze
WaadMahadsantahay
Rahmat
Tanemirt
Tenki
Chnorakaloutioun
TänanVäga
KhobChaiDeu
Ačiū
Barkal
Aabhar
Dhanyavaad
Grazzi
Nouari
Mauruuru
GoRaibhMaithAgat
KyayTzuTinPaTe
Mochchakkeram
KurreSumanga
Yekeniele
Dhanyabaad
KopKhunKrap
Tau
Nanni
Milesker
Gratzias
Trugéré
Akpé
Misaotra
Tanmirt
Akiba
Gràçie
MaloTeOfa
Asante
Miigwetch
TakkFyri
Díky
Marahaba
DankJuWel
Graciñas
Mahalo
Wado
Spas
Diolch
EskerrikAsko
Dankawol
Dankie
Paldies
DjiereDieuf
ANiKié
Rahmet
TeşekkürEderim
Sobodi
AabhariAahe
Trugarez
Bedankt
Zikomo
Enkosi
M-Sapo
FaafetaiLava
Ngiyabonga
TapadhLeibh
Shukriya
AsanteSana
Waita
Welalin
TangioTumas
Dhanyavad
Sukriya
Tänan
Mamnun
TesekkurEdirem
KeYaLeboha
Vinaka
Barka
Dankon
aNiKié
Toda
Mersi
Nizzik ajr
Aguyjé
Dhanyavadagalu
Dhanyavadalu
Sulpáy
Blagodaria
Nandri
Merkzi
Akun
ÇoxSagOlun
Meharbani