
Rest here

Goals

Not programming

Architecture and Design

REST

REpresentational **S**tate **T**ransfer

Introduction to REST

Key drivers

Architectural properties

What is REST ?

Richardson Maturity Model

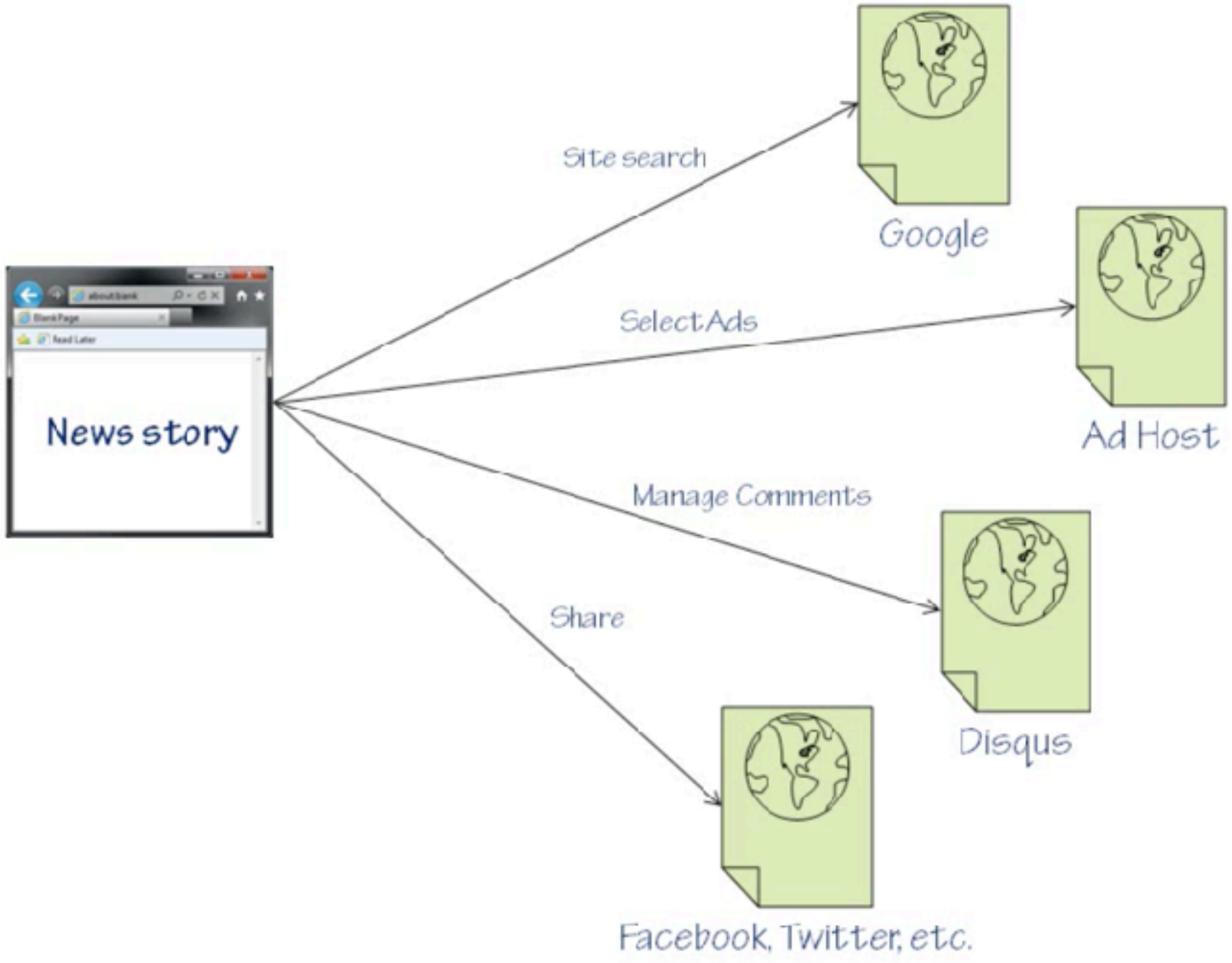
Key drivers of REST

Heterogeneous Interoperability

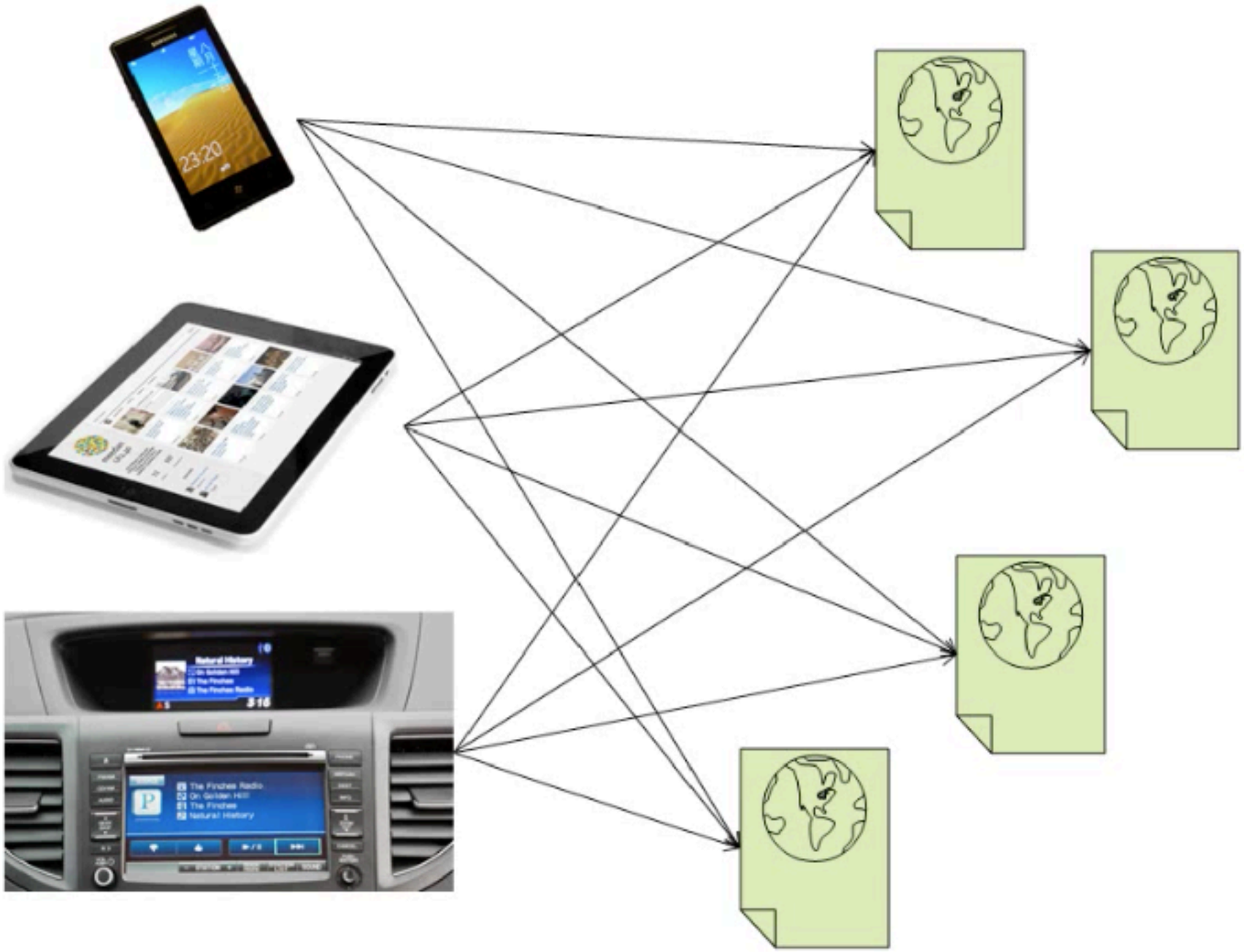
Devices

Cloud

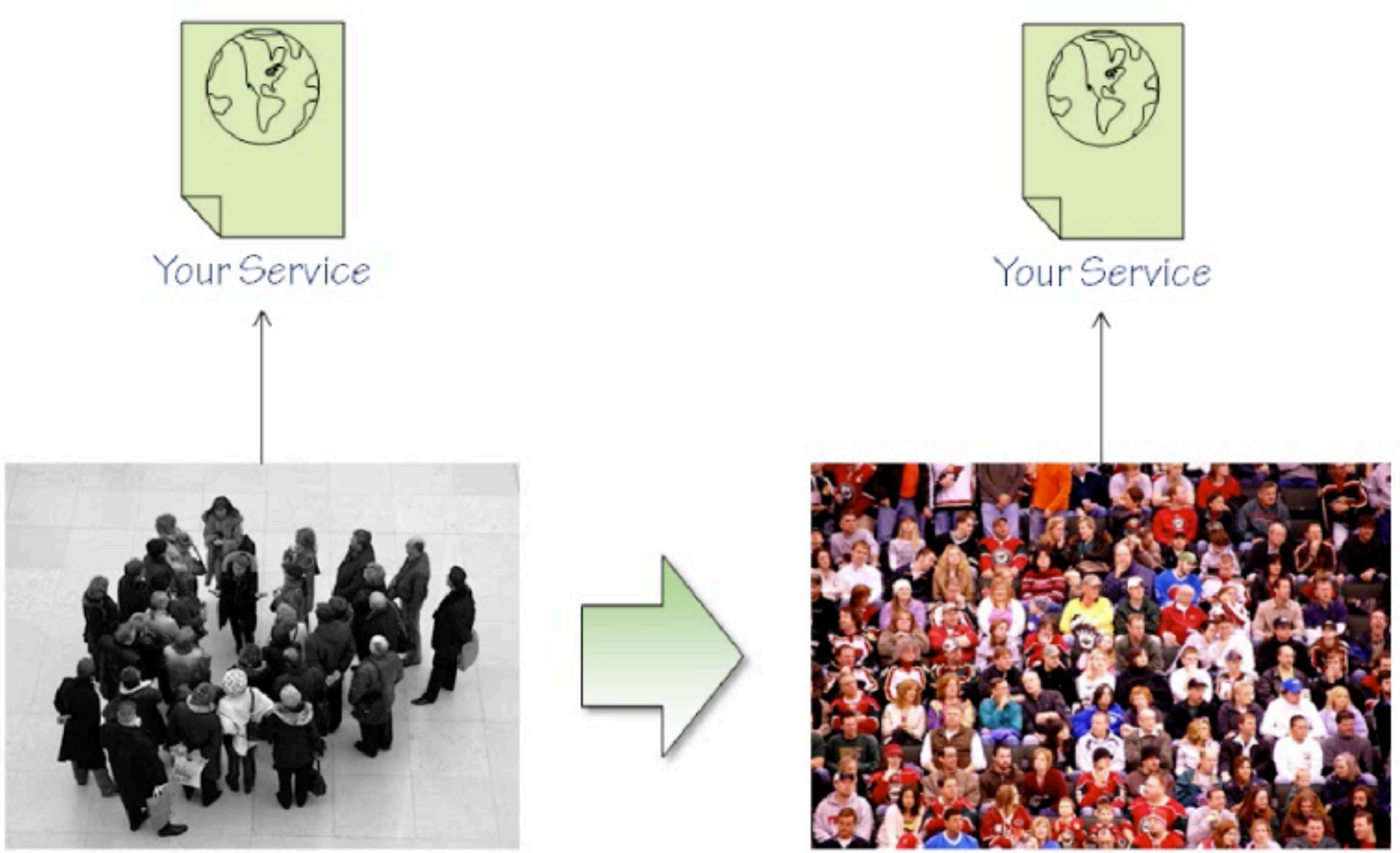
Heterogeneous Interoperability



Devices

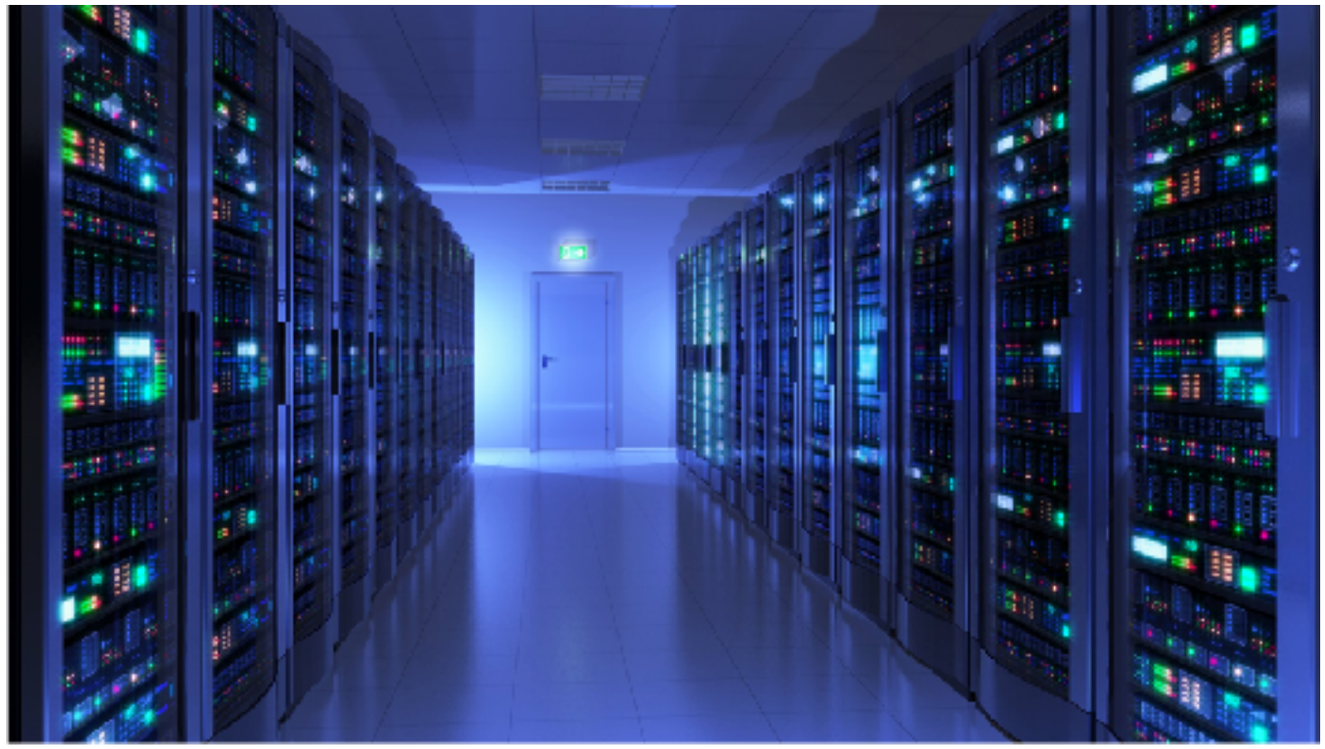


Cloud



“It is critical to build a scalable architecture in order to take advantage of a scalable infrastructure”

Move to Cloud



Properties of REST

Heterogeny

Scalability

Evolvability

Visibility

Reliability

Efficiency

Performance

Manageability

Quiz

Heterogeny

Scalability

Evolvability

Visibility

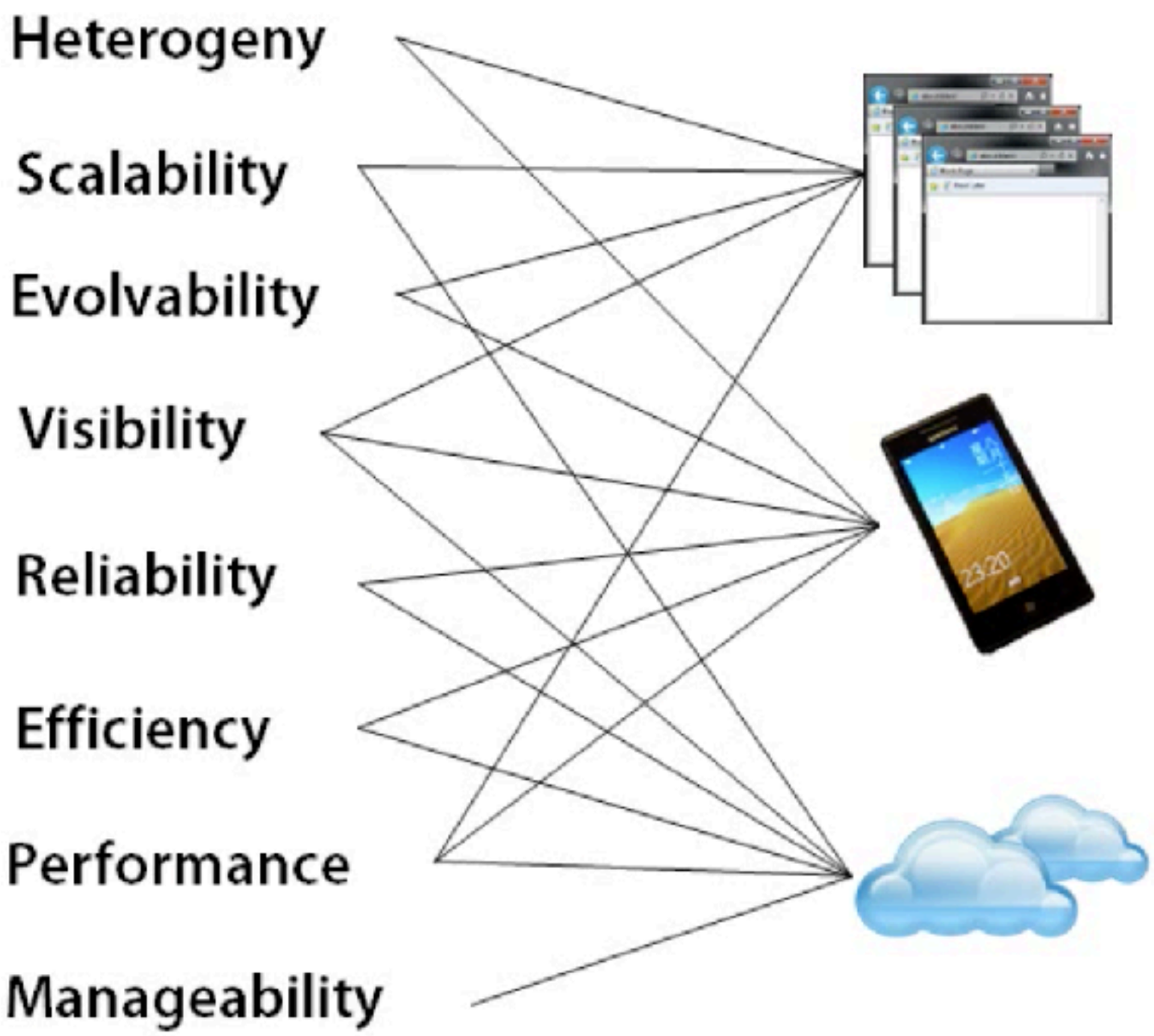
Reliability

Efficiency

Performance

Manageability

Properties of REST



What is REST ?

Architectural style

First introduced in 2000

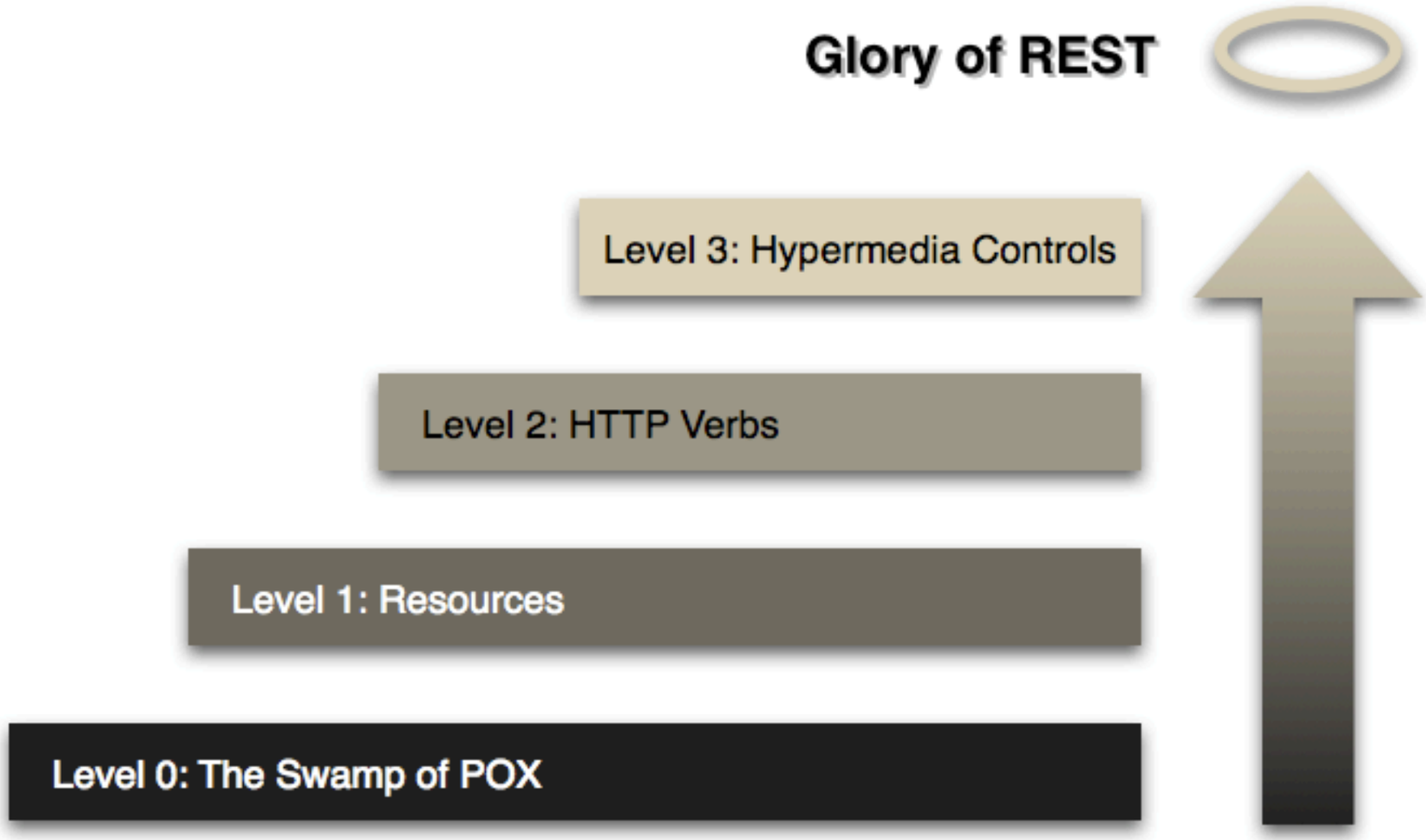
What is not REST ?

Protocol

RPC (Remote Procedure Call)

HTTP

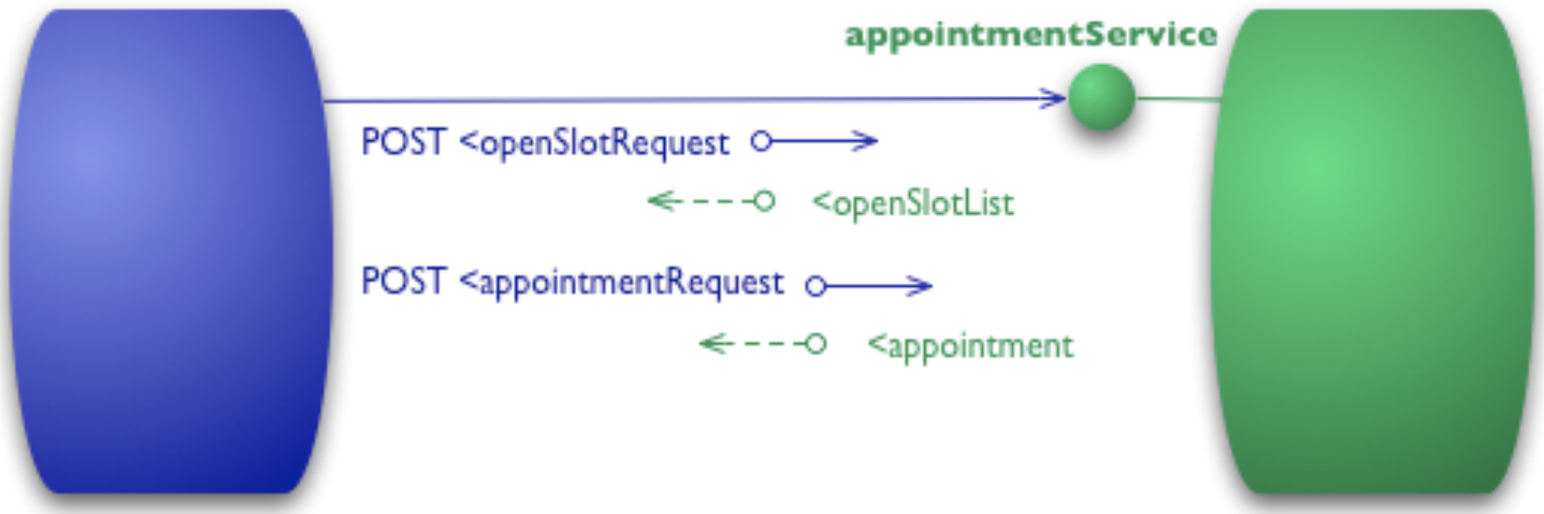
Richardson Maturity Model



https://martinfowler.com/articles/richardsonMaturityModel.html

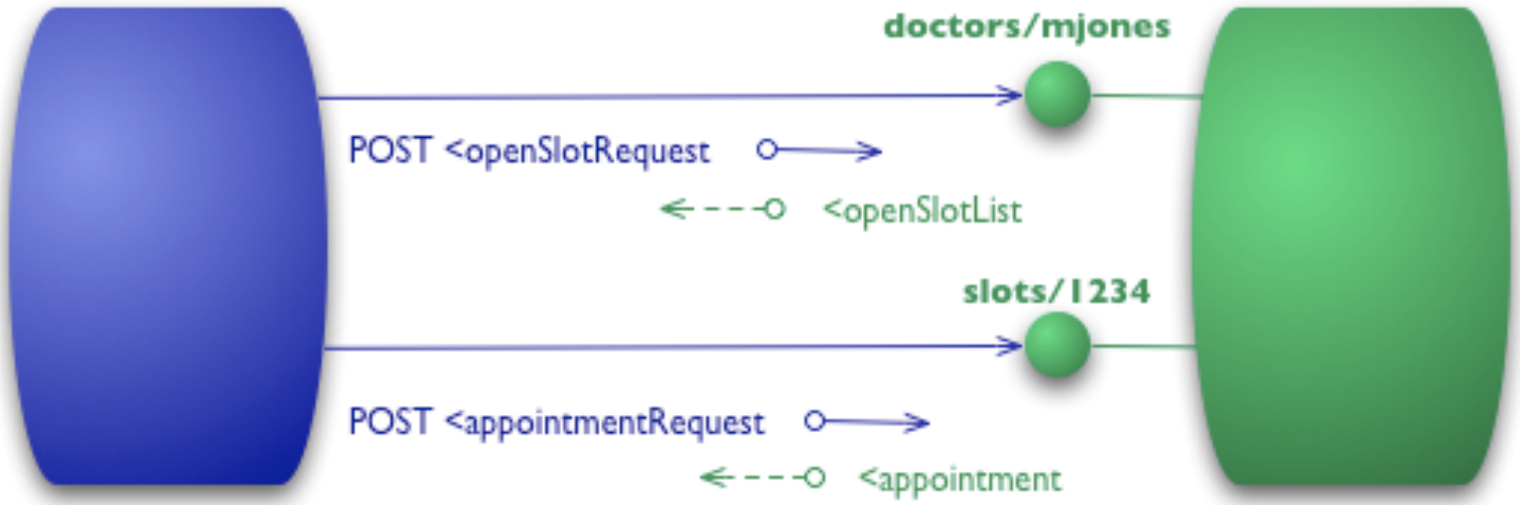
Level 0

Remote Interaction mechanism



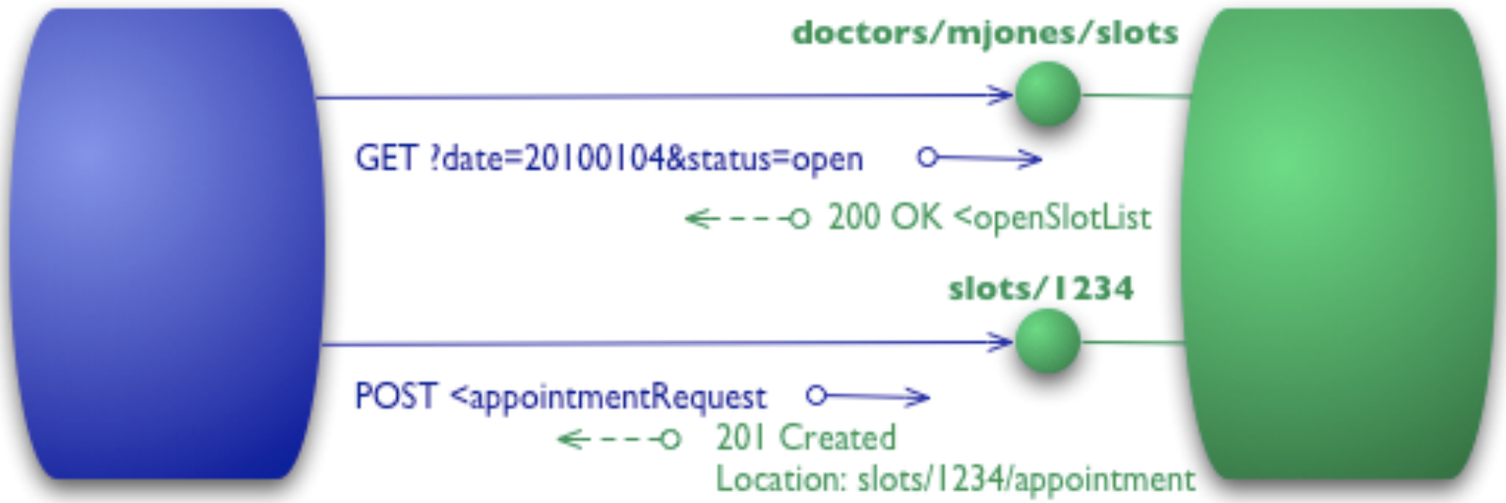
Level 1 :: Resources

Individual resources



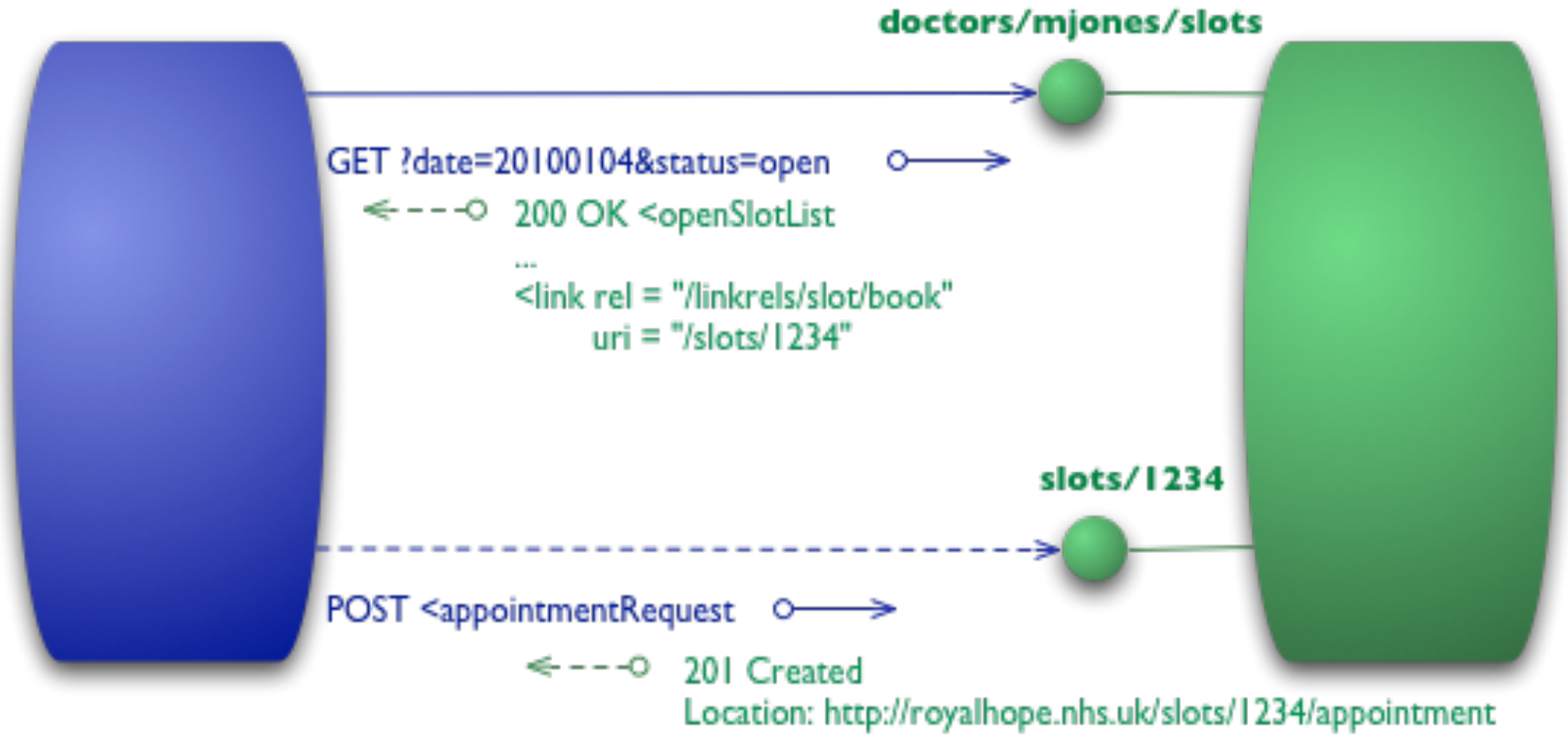
Level 2 :: HTTP verbs

Interaction through HTTP



Level 3 :: Hypermedia controls

Self-Document



Summary

Level 1

breaking a large service

into multiple resources

Summary

Level 2

Standard set of verbs

Summary

Level 3

Discoverability, provide a way of making a protocol more self-docsmeting

REST constraints

REST constraints

Client-server

Stateless

Cache

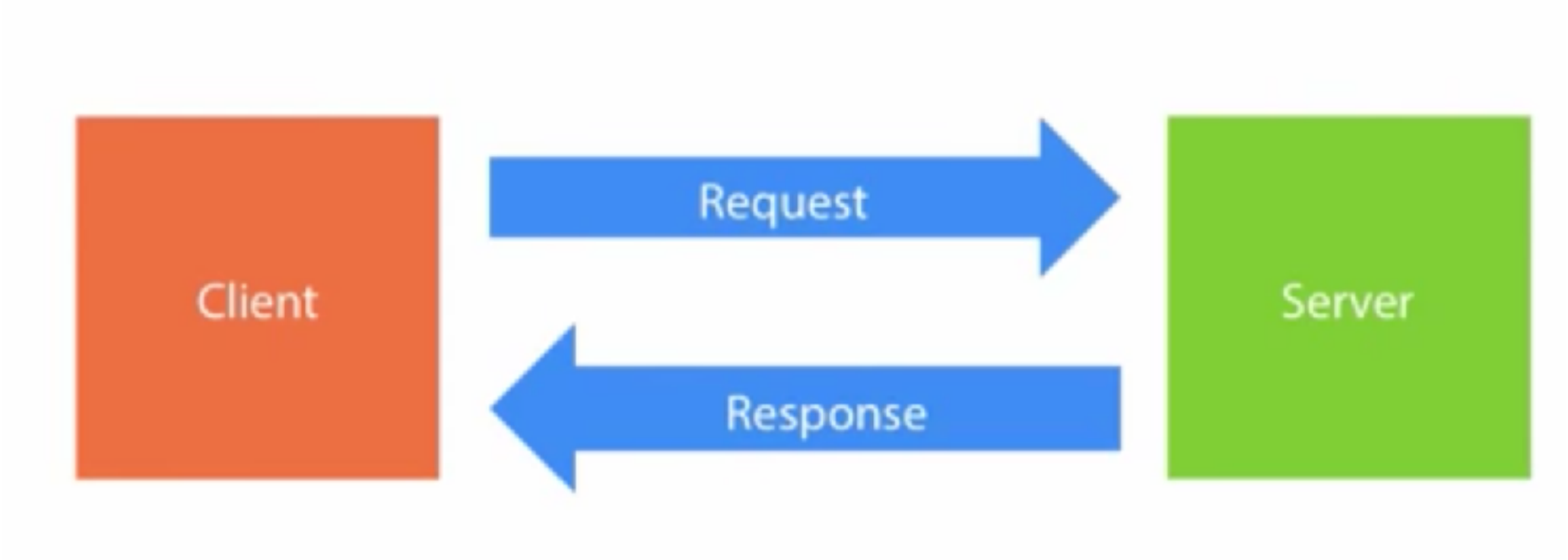
The Uniform Interface

Layered system

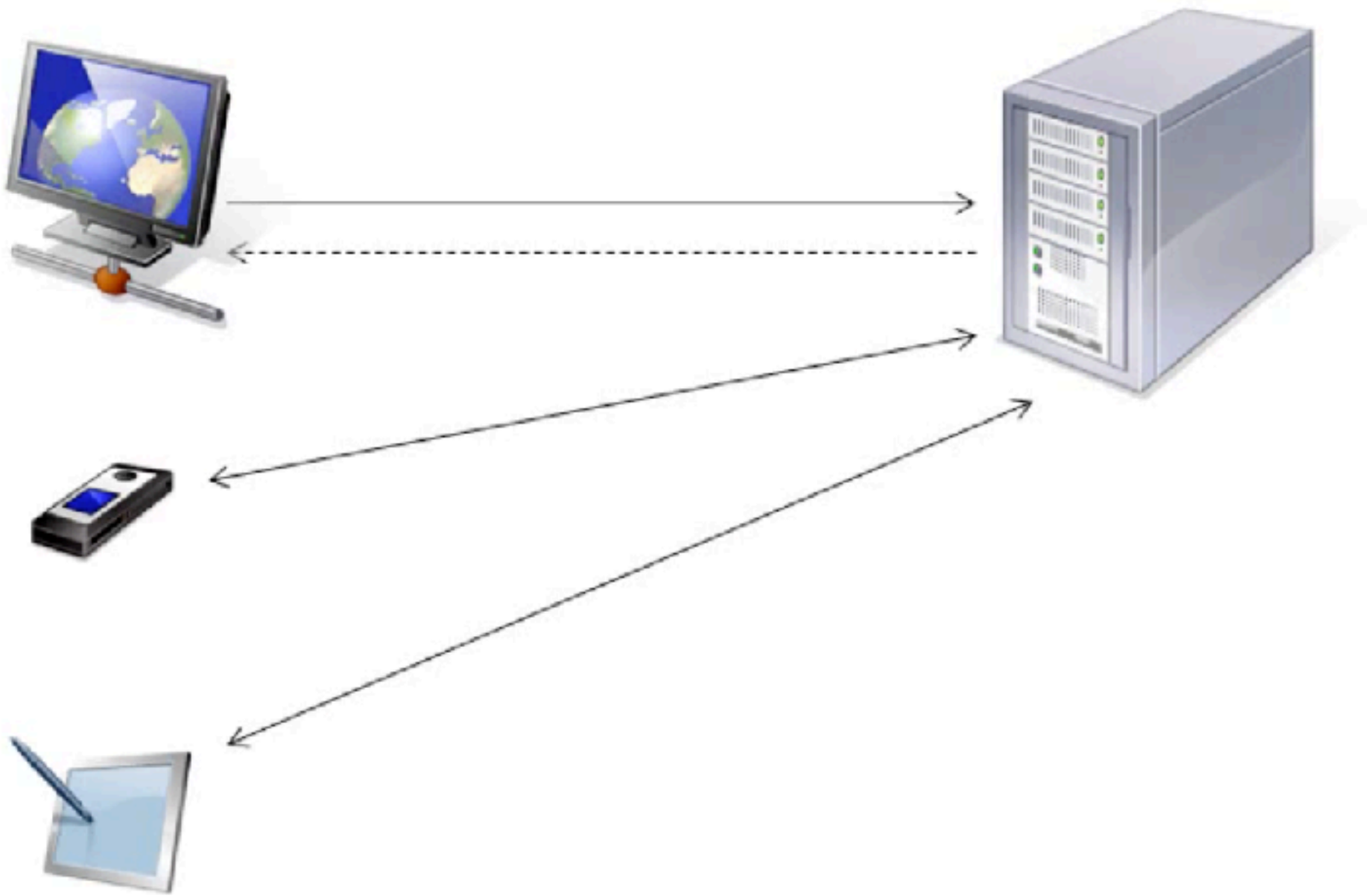
Code-on-Demand

Client-server

Client-server



Client-server



Concerns

Security

Administration

Heterogeneous network

Complexity

Benefits

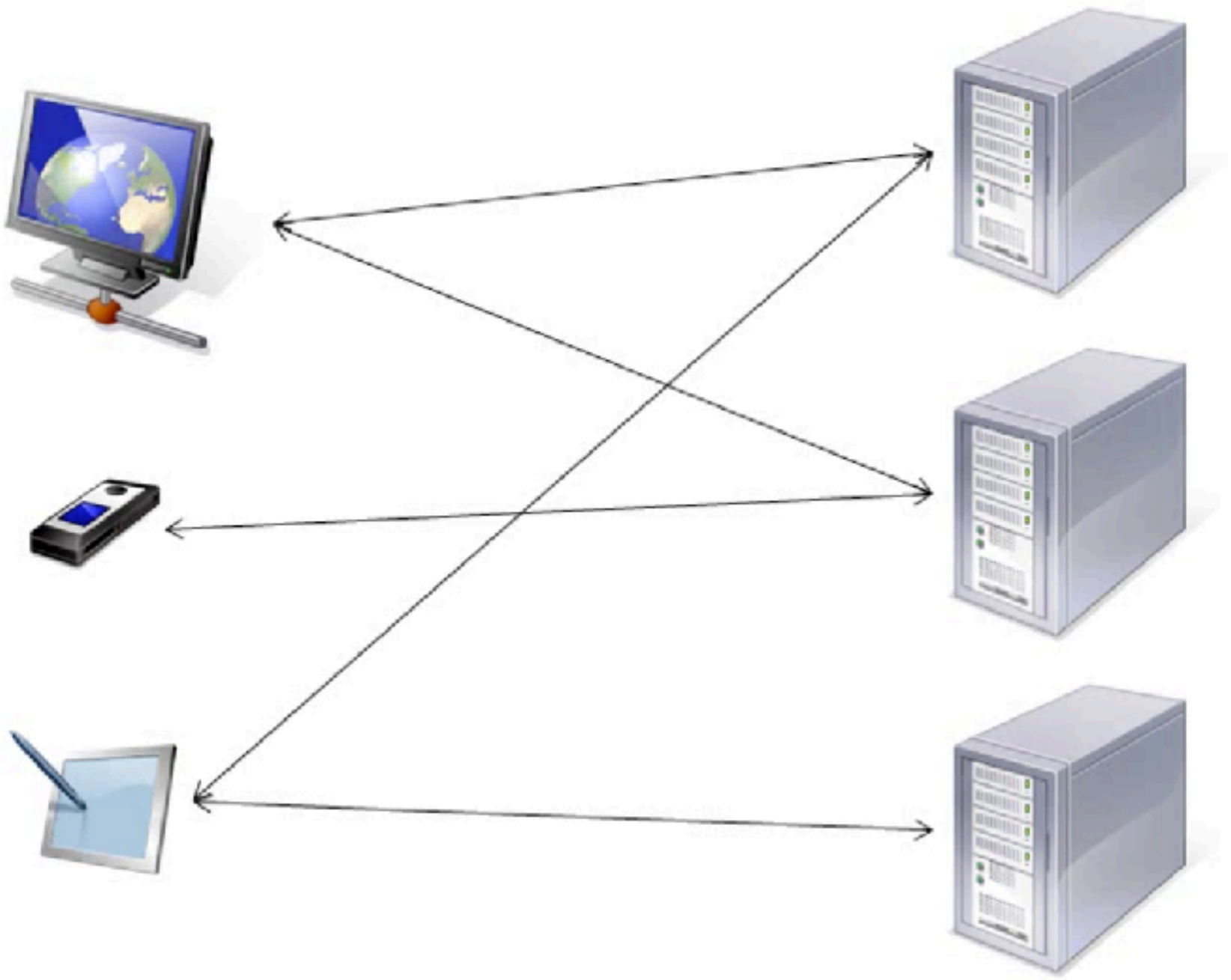
Portability of clients

Scalability

Evolvability

Stateless

Stateless



Concerns

Network reliability

Network topology

Complexity

Administration

Benefits

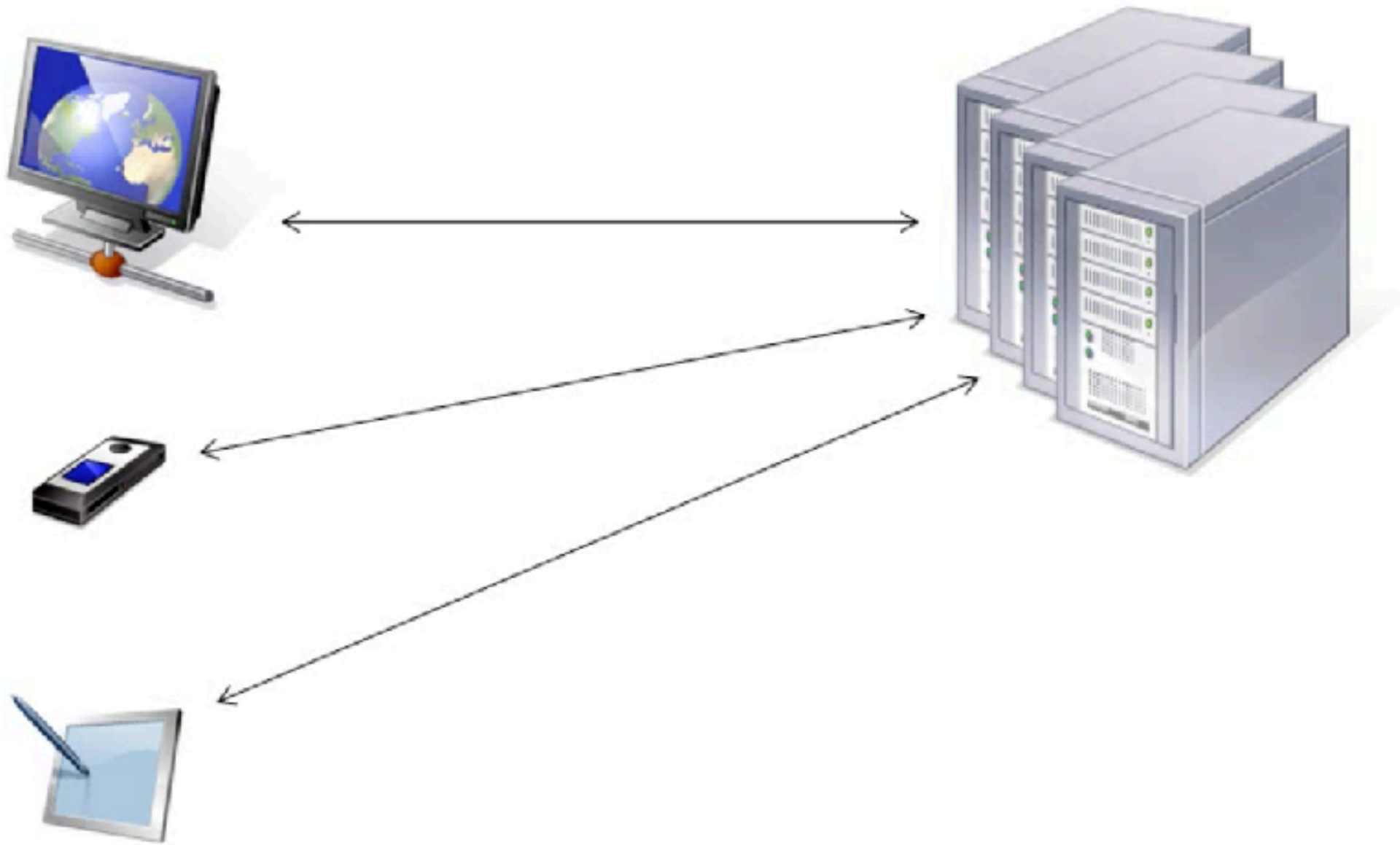
Visibility

Reliability

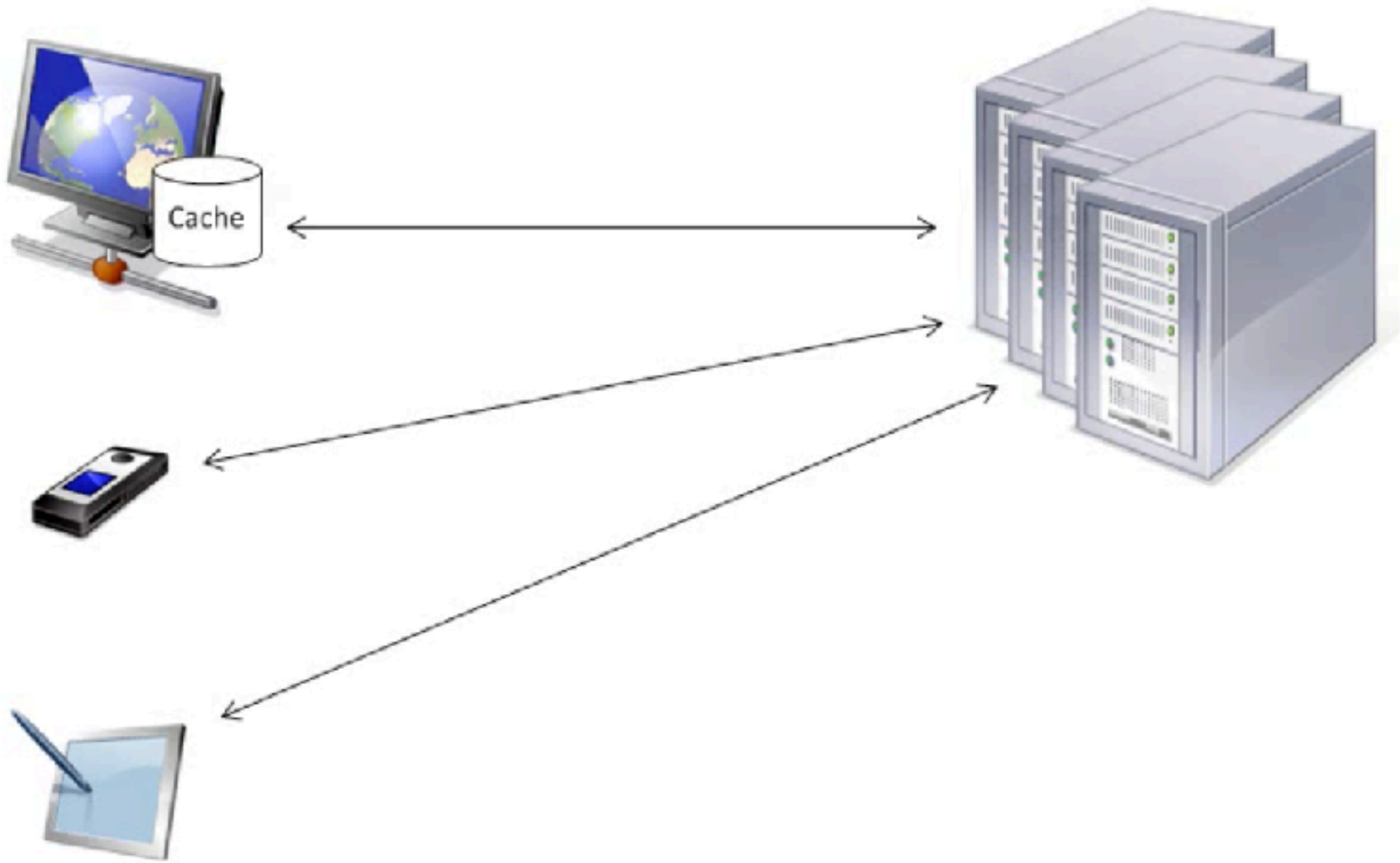
Scalability

Cache

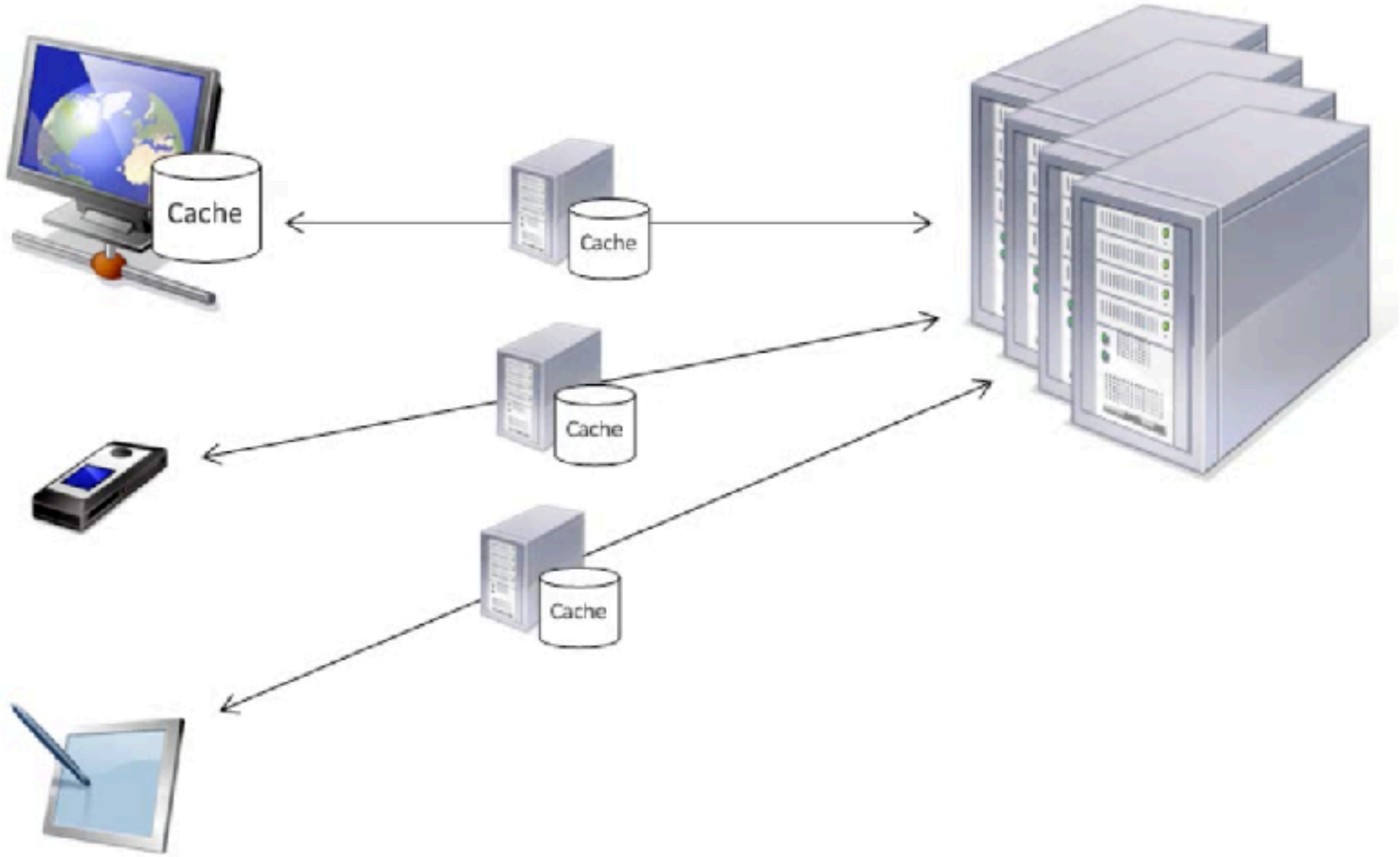
Cache



Cache



Cache



Concerns

Latency

Bandwidth

Transport cost

Benefits

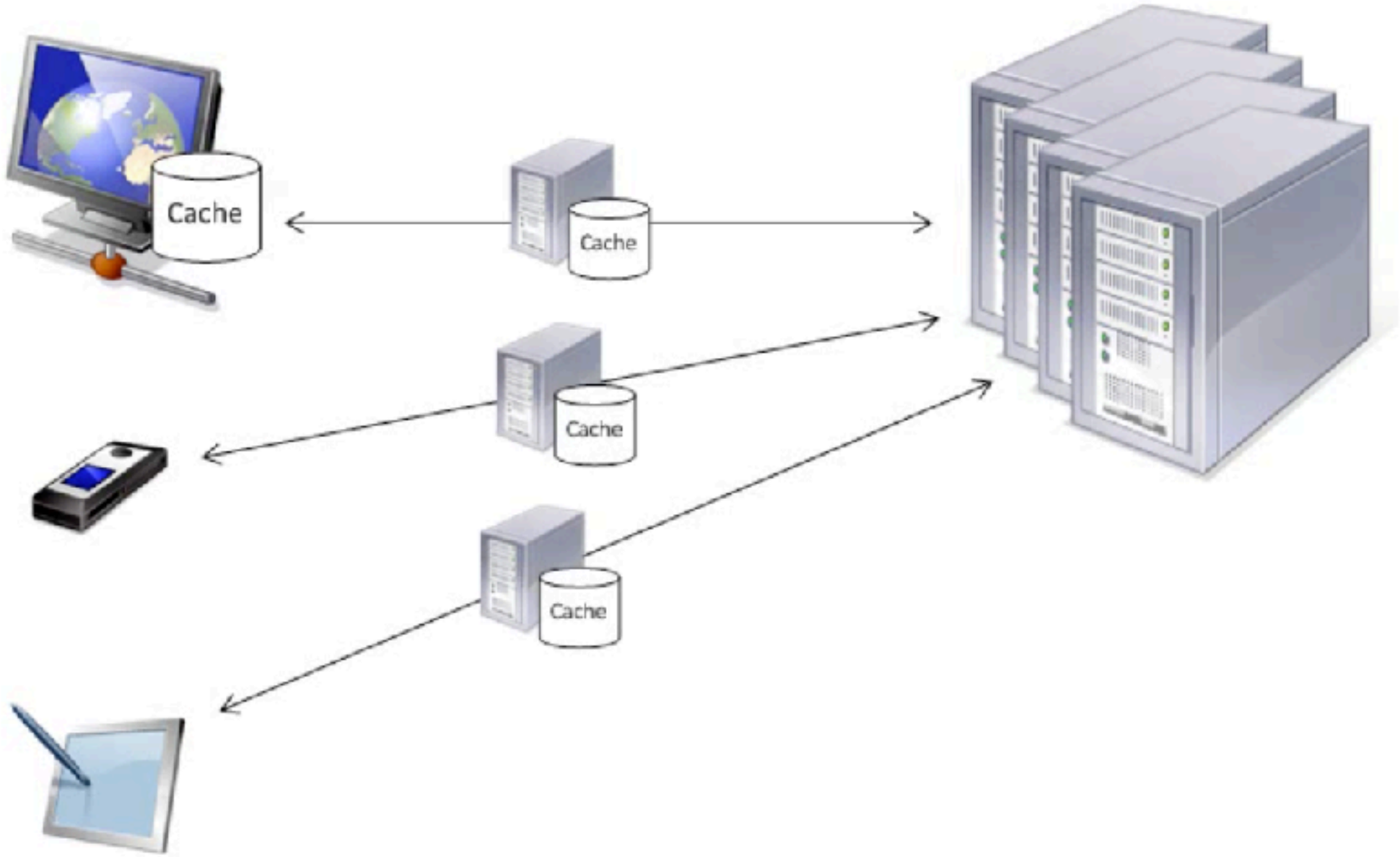
Efficiency

Scalability

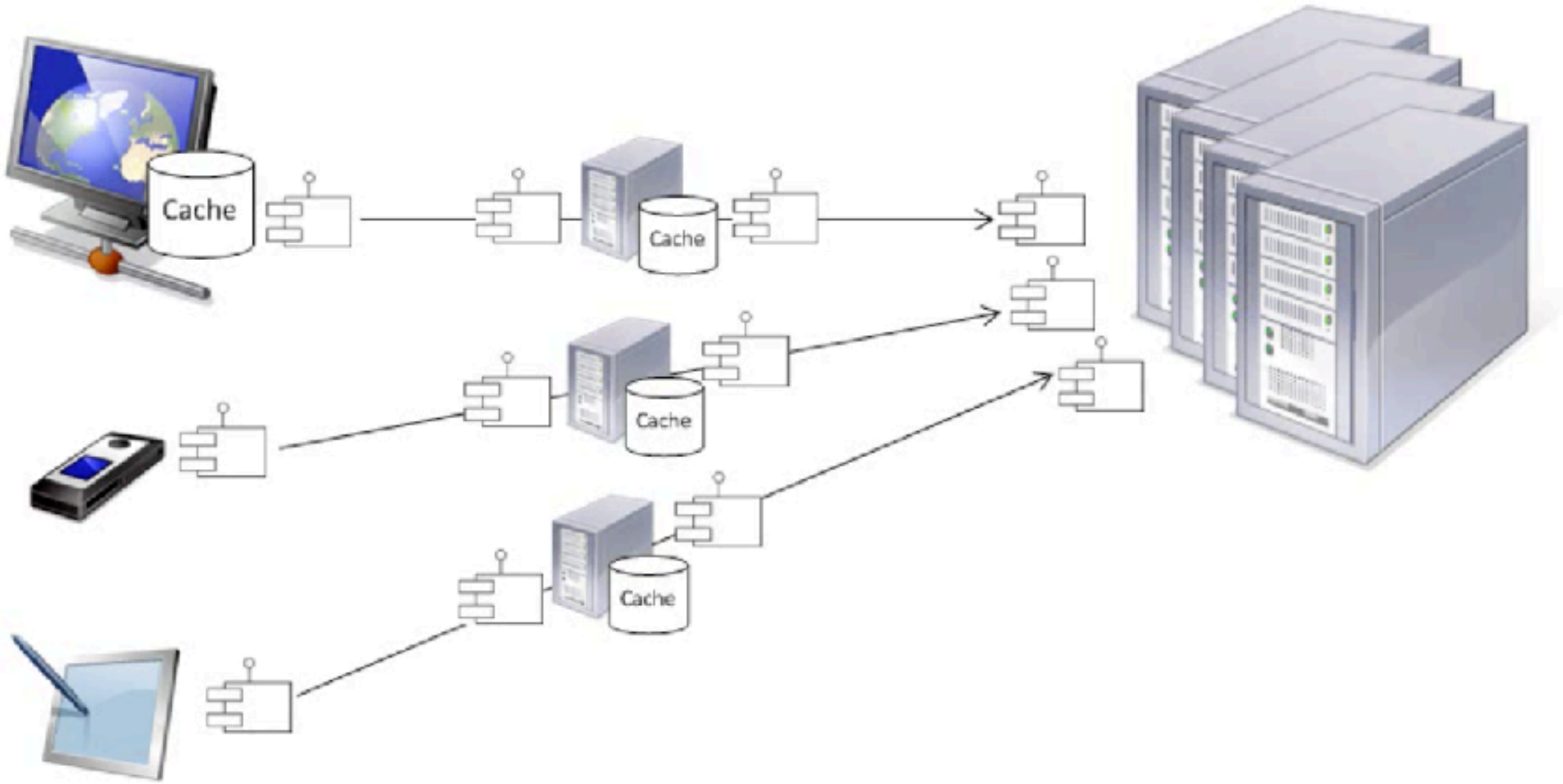
Performance

The Uniform Interface

The Uniform Interface



The Uniform Interface



Elements of the Uniform interface

Identification of **resources**

Manipulation through **representations**

Self-descriptive messages

Hypermedia as the engine of app state

Concerns

Network reliability

Network topology

Administration

Complexity

Heterogeneous network

Benefits

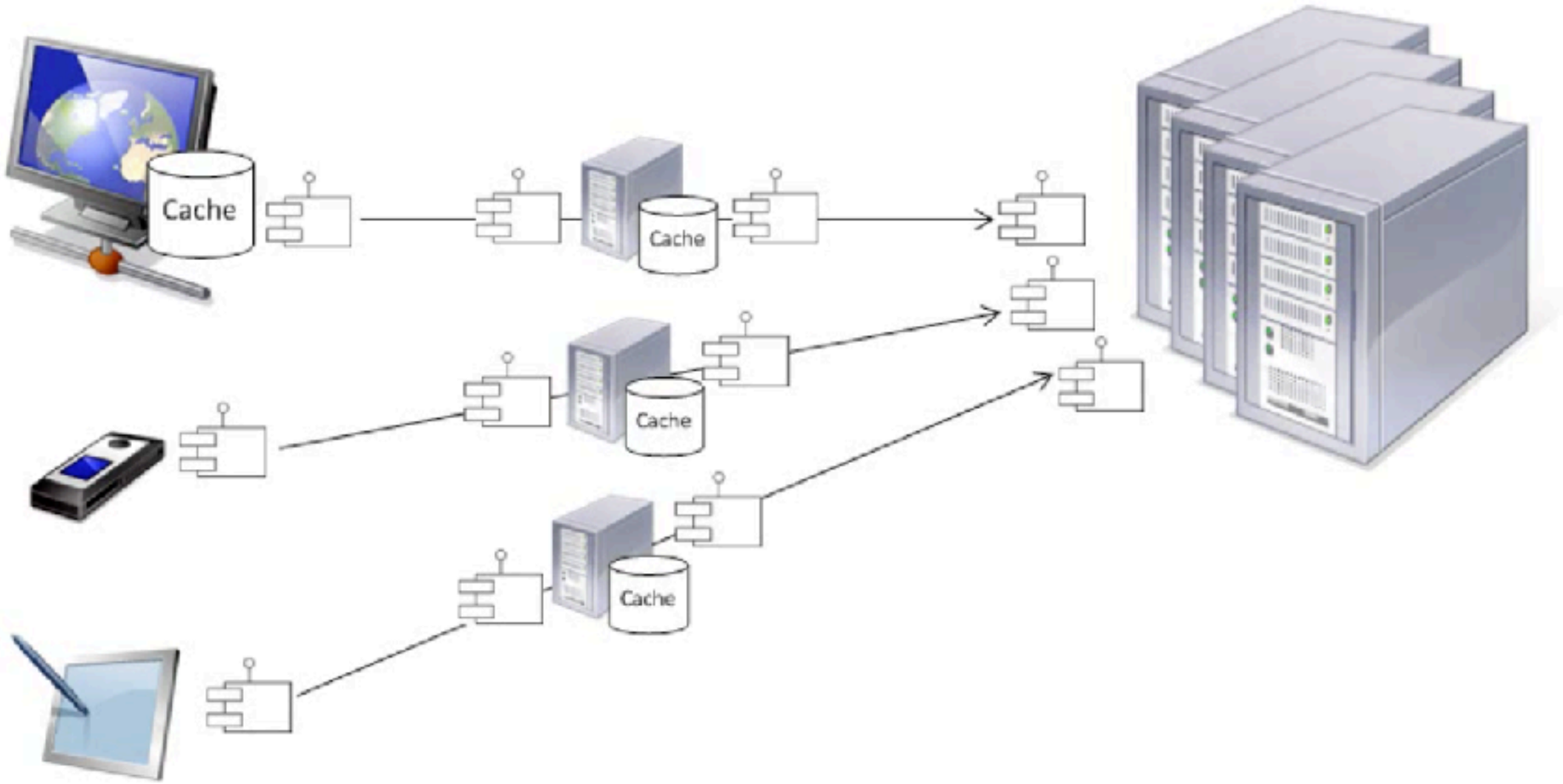
Visibility

Evolvability

Performance

Layered system

Layered system



Concerns

Network topology

Complexity

Security

Benefits

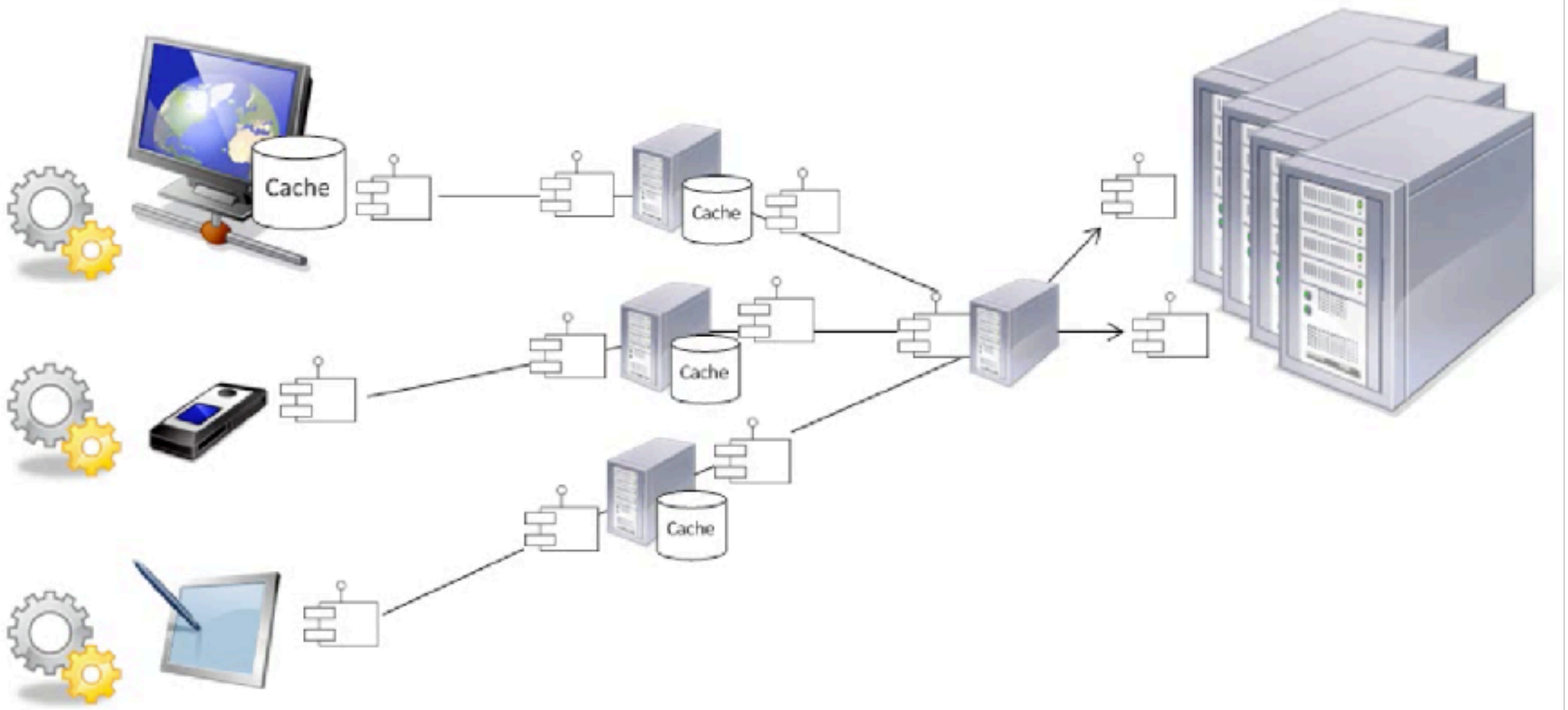
Scalability

Manageability

Performance

Code-on-demand (optional)

Code-on-demand



Code-on-demand

Help to manage complexity

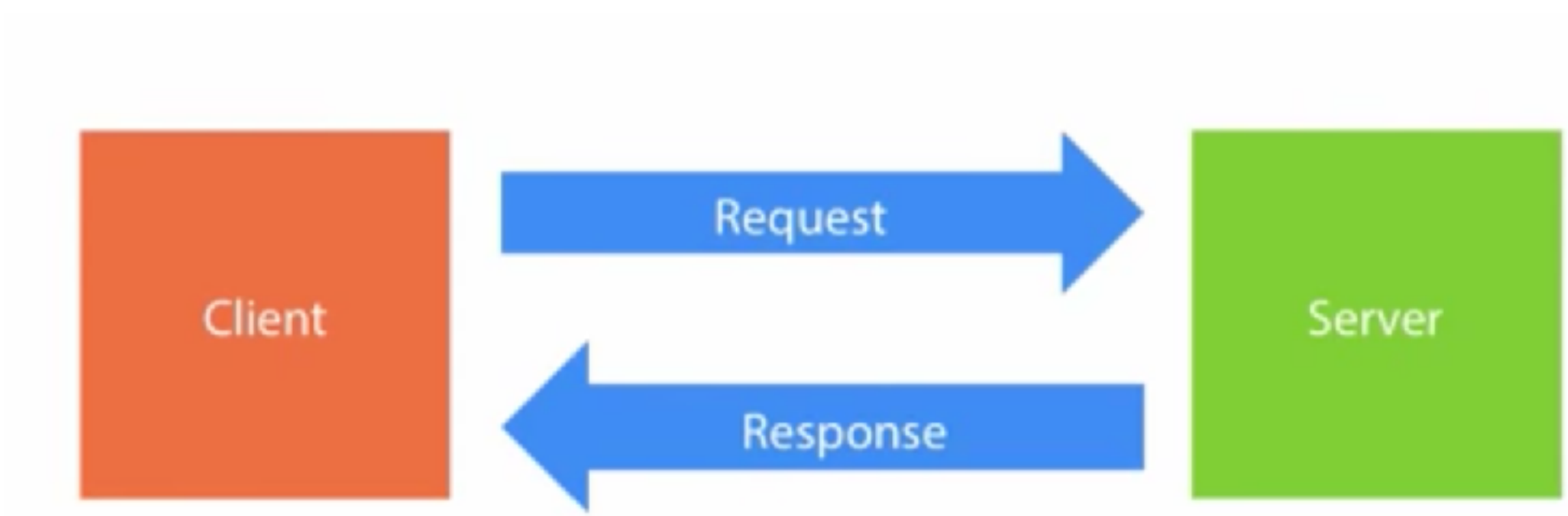
The trade-off is visibility

Designing REST

Designing REST

For services

For client



RESTFul

POST : Add data

GET : Retrieve data

DELETE : Delete data

PUT : Update data

HTTP Code

Code	Name	Description
200	OK	Everything is working
201	Created	New resource has been created

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

HTTP Code

Code	Name	Description
301	Moved Permanently	
302	Found	Temporary redirect
304	Not Modified	The client can use cached

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

HTTP Code

Code	Name	Description
400	Bad Request	Everything is working
401	Unauthorized	New resource has been created
403	Not Modified	The client can use cached data
404	Not found	There is no resource behind the URI
422	Unprocessable Entity	

https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

HTTP Code

Code	Name
500	Internal Server Error
502	Bad Gateway
503	Service Unavailable

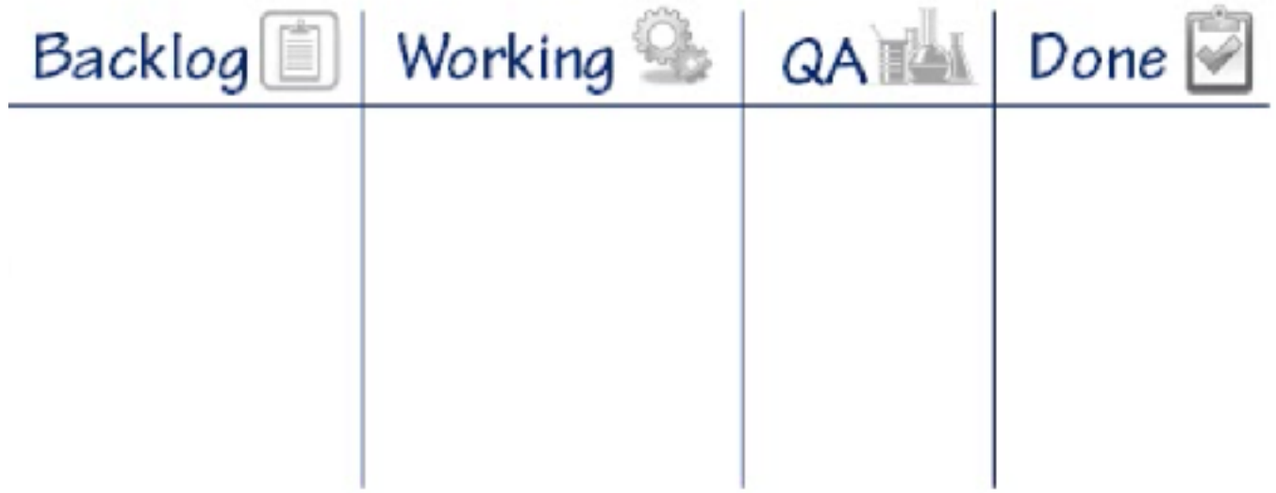
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

Example

Bug tracking

Bug

Bug board



Bug tracking

Stories	To Do	In Progress	Testing	Done
This is a sample text. Replace it with your own text.	This is a sample text. Replace it with your own text. This is a sample text. Replace it with your own text. This is a sample text. Replace it with your own text. This is a sample text. Replace it with your own text.	This is a sample text. This is a sample text. This is a sample text.	This is a sample text. This is a sample text. This is a sample text.	This is a sample text. Replace it with your own text. This is a sample text. Replace it with your own text.
This is a sample text. Replace it with your own text.	This is a sample text. This is a sample text. This is a sample text. This is a sample text.	This is a sample text. This is a sample text. Replace it with your own.	This is a sample text. This is a sample text.	This is a sample text. Replace it with your own text.

Let’s start to design service

Step to design service

List of **requirements**

Identify the **state transitions**

Identify the **resources**

Design the **media types**

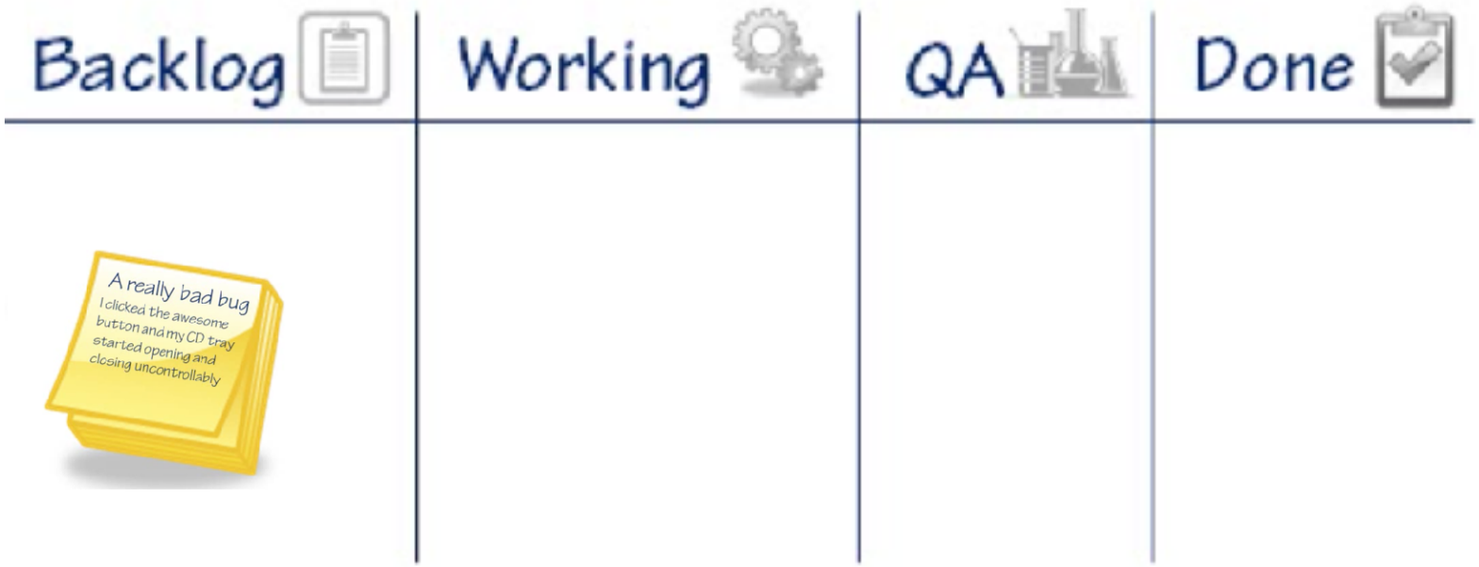
1. Requirements

Bugs in each workflow states

1. Backlog
2. Working
3. QA/Testing
4. Done

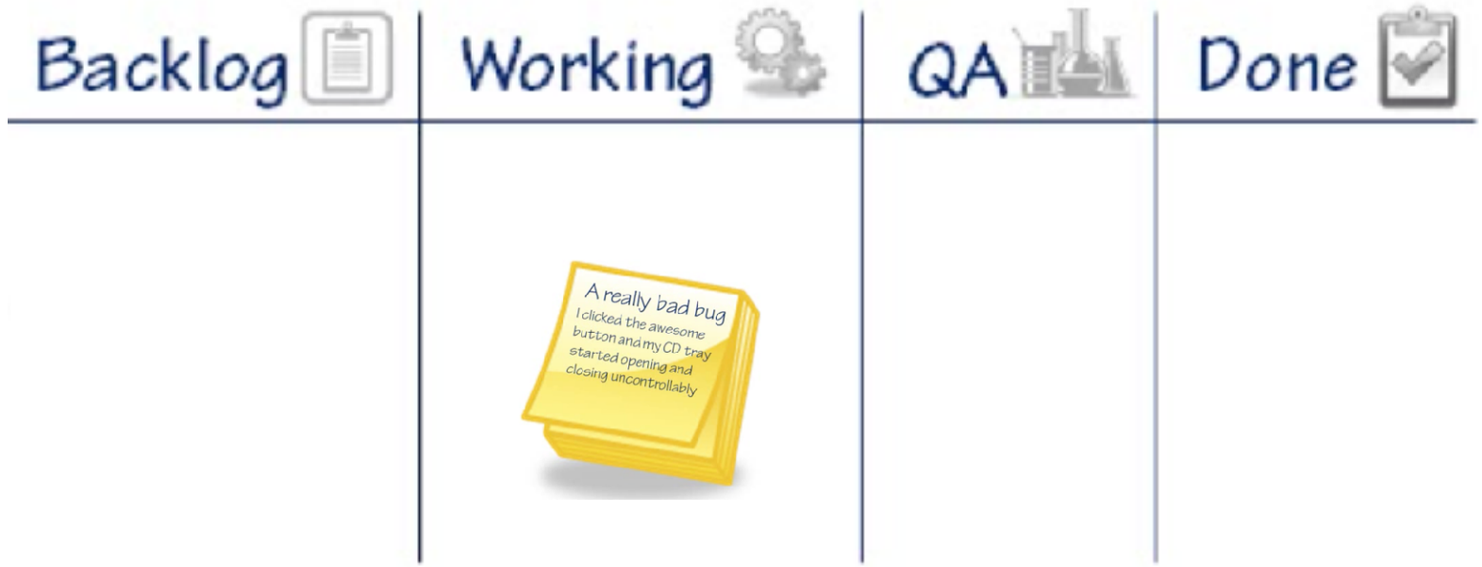
1. Requirements

Add a new bug to the backlog



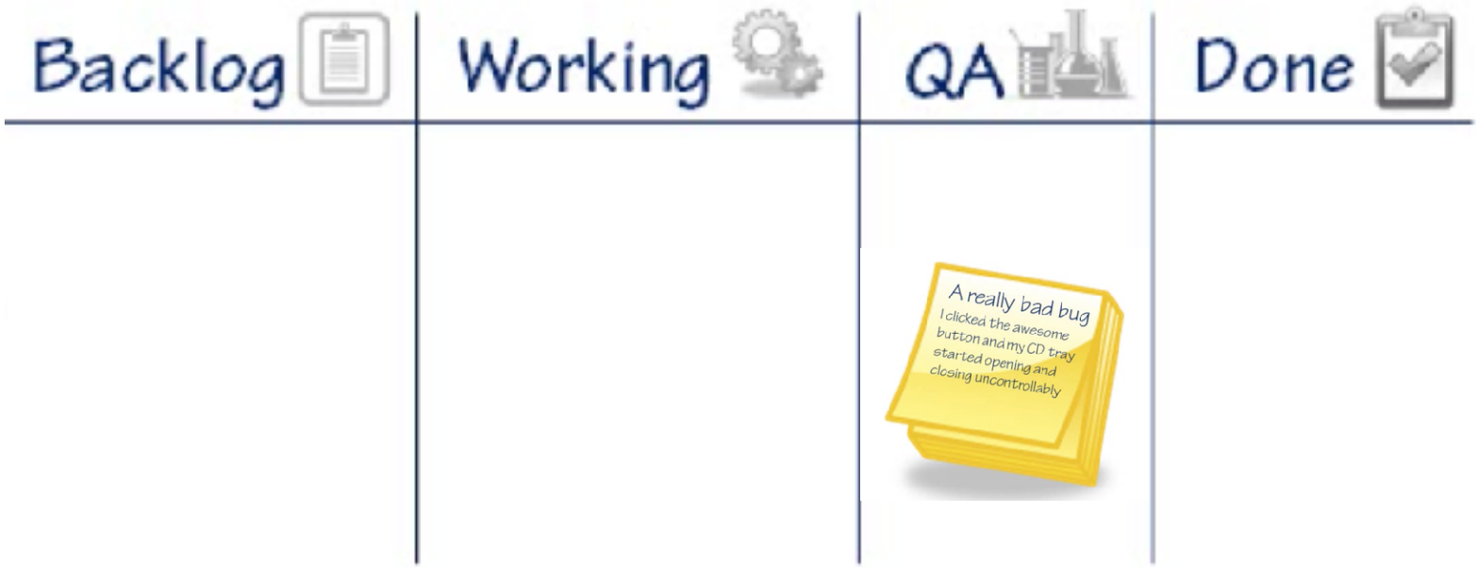
1. Requirements

Move a bug to other state



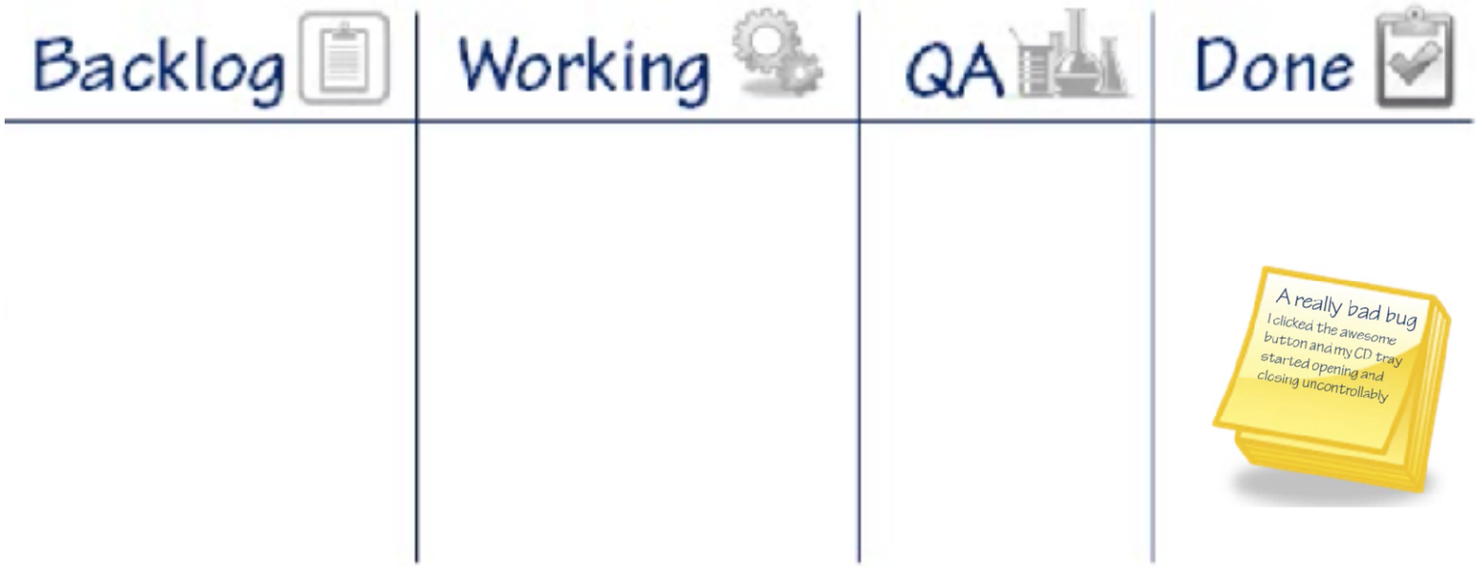
1. Requirements

Move a bug to other state

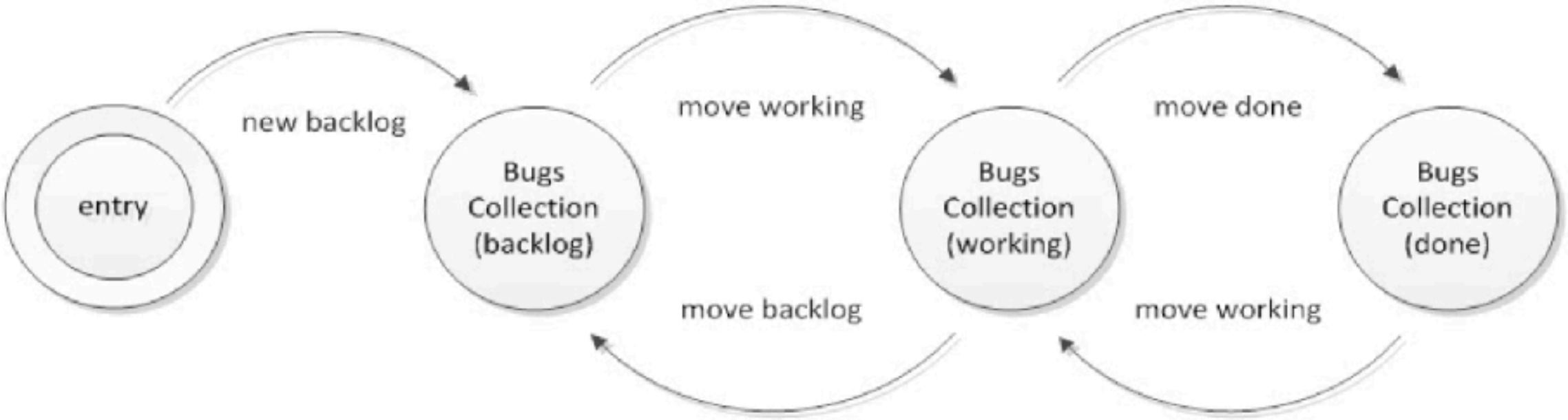


1. Requirements

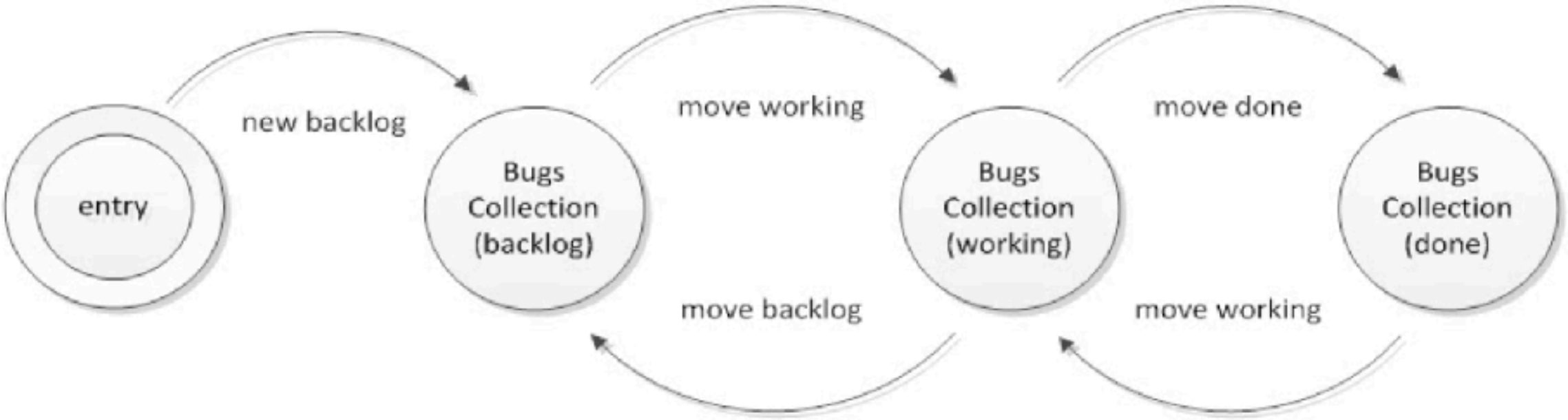
Complete a bug



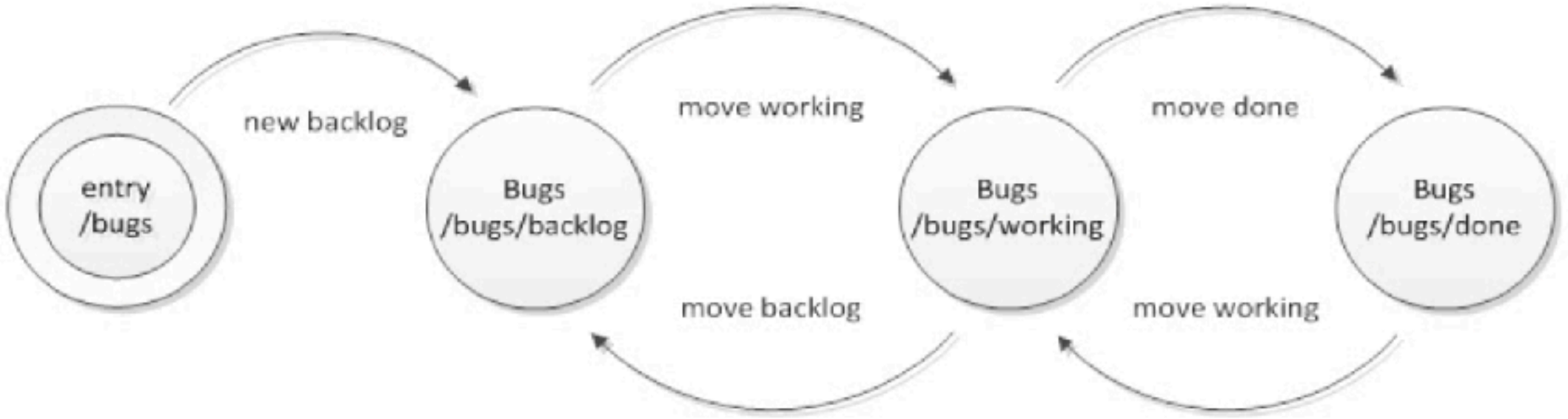
2. Application state transition



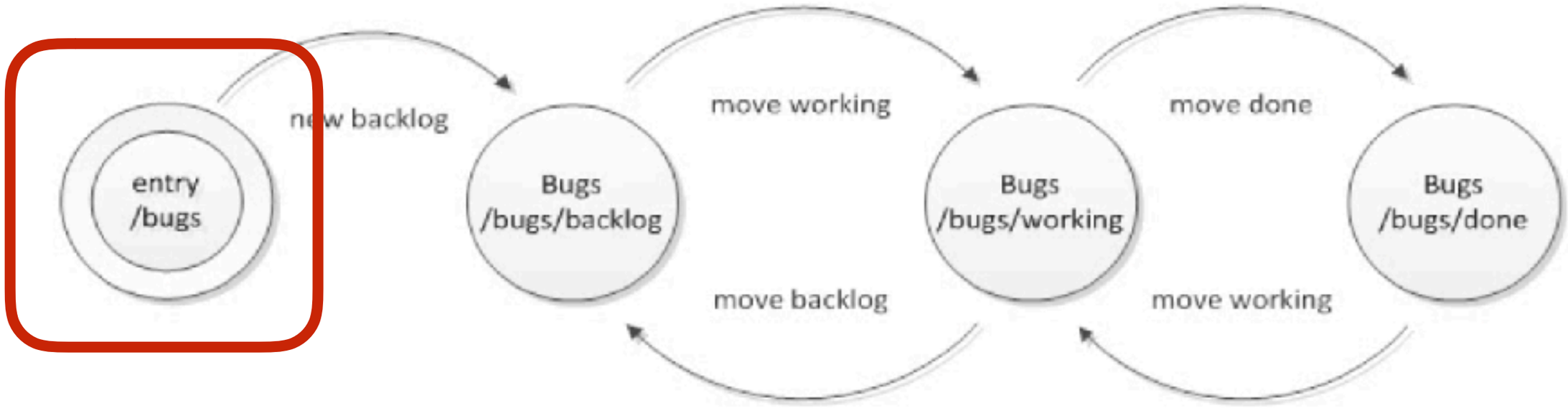
3. Identify the resources



3. Identify the resources



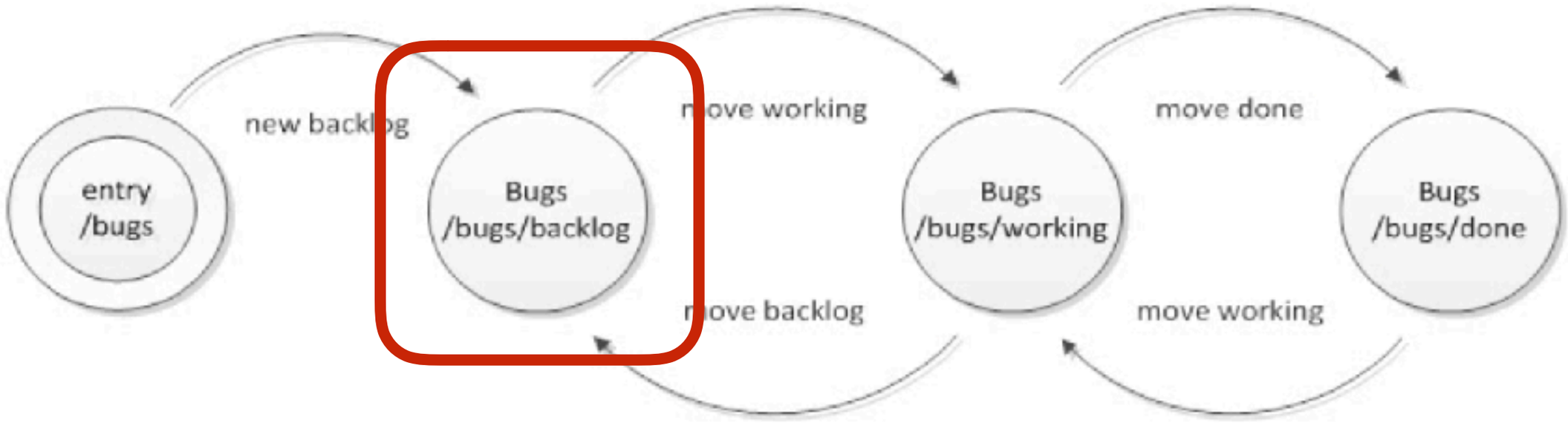
3. Identify the resources



/bugs is entry point

Resources	HTTP Method	Description
/bugs	GET	List of bugs
/bugs/:id	GET	Get bug information
/bugs	POST	Create new bug

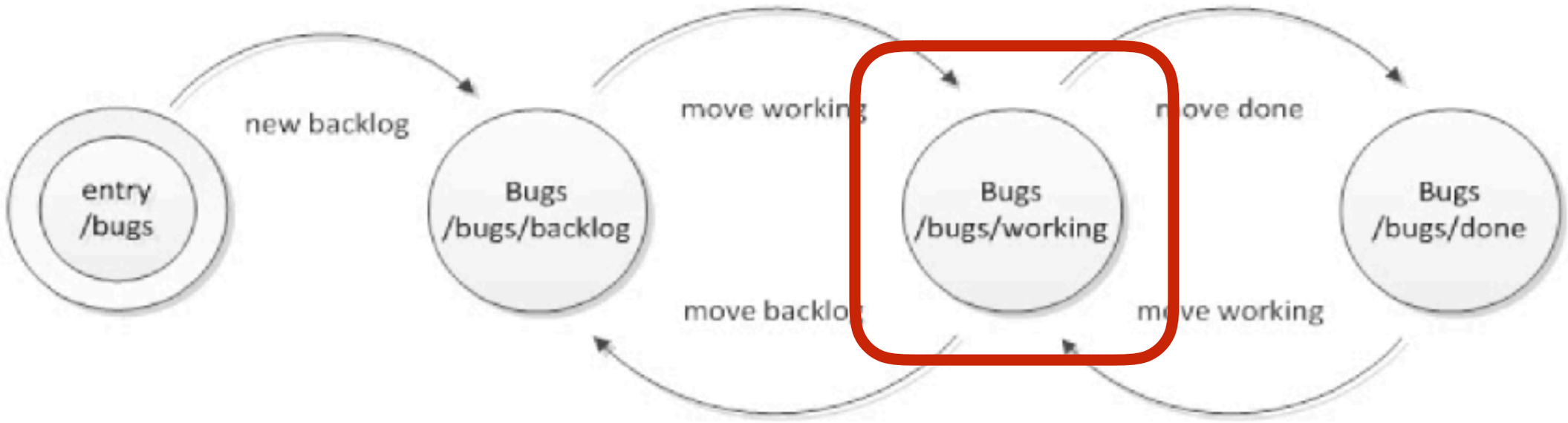
3. Identify the resources



/bugs/backlog

Resources	HTTP Method	Description
/bugs/backlog	GET	List of bugs in backlog
/bugs/backlog	POST	Add new bug to the backlog

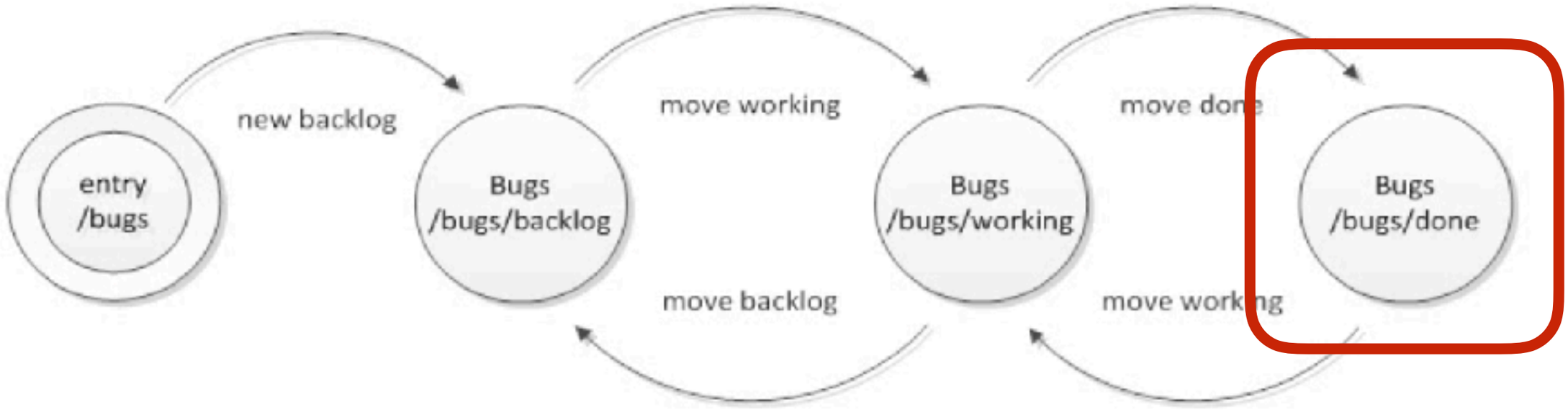
3. Identify the resources



/bugs/working

Resources	HTTP Method	Description
/bugs/working	GET	List of bugs in working
/bugs/working	POST	Activate a bug

3. Identify the resources



/bugs/done

Resources	HTTP Method	Description
/bugs/done	GET	List of bugs in done
/bugs/done	POST	Complete a bug

4. Design media types

TEXT, HTML, XML, **JSON** and etc.

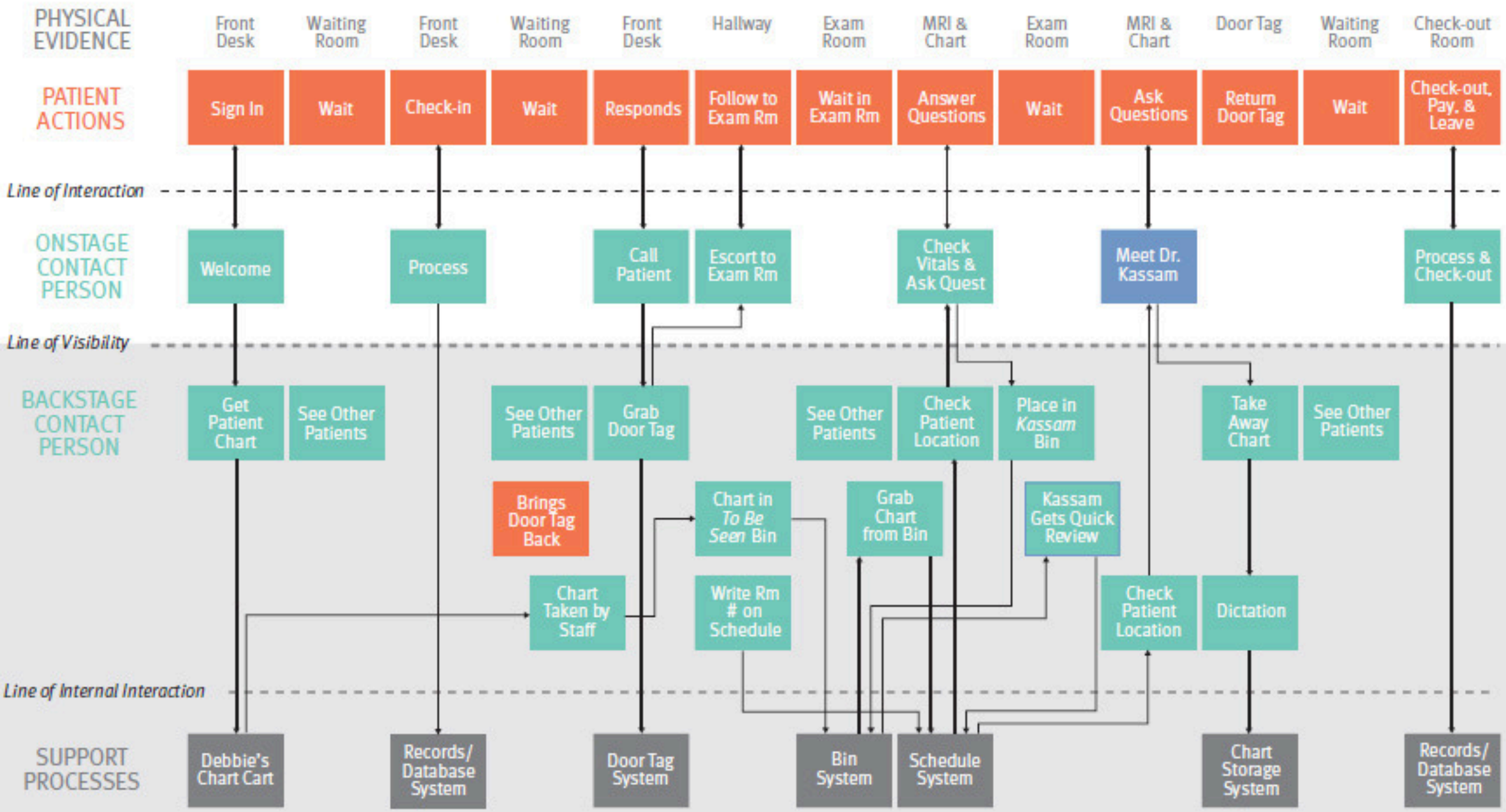
```
{
  "id": 1,
  "title": "Bug 01",
  "description": "Bad Bug",
  "status": "new|backlog|working|done"
}
```

4. Design media types

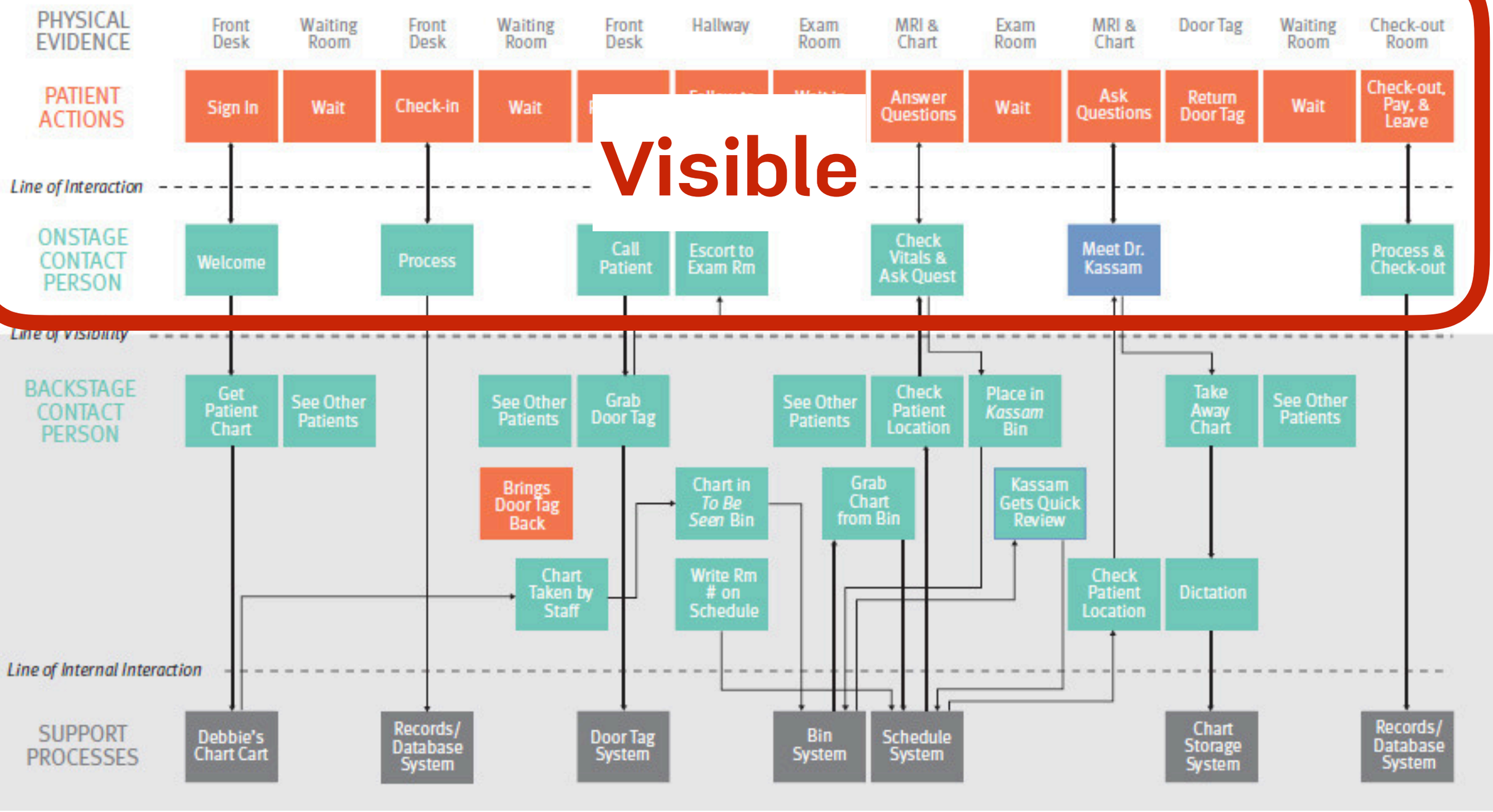
```
[
  {
    "id": 1,
    "title": "Bug 01",
    "description": "Bad Bug",
    "status": "new"
  },
  {
    "id": 2,
    "title": "Bug 02",
    "description": "Bad Bug",
    "status": "backlog"
  },
  {
    "id": 3,
    "title": "Bug 03",
    "description": "Bad Bug",
    "status": "working"
  }
]
```

Service design blueprint

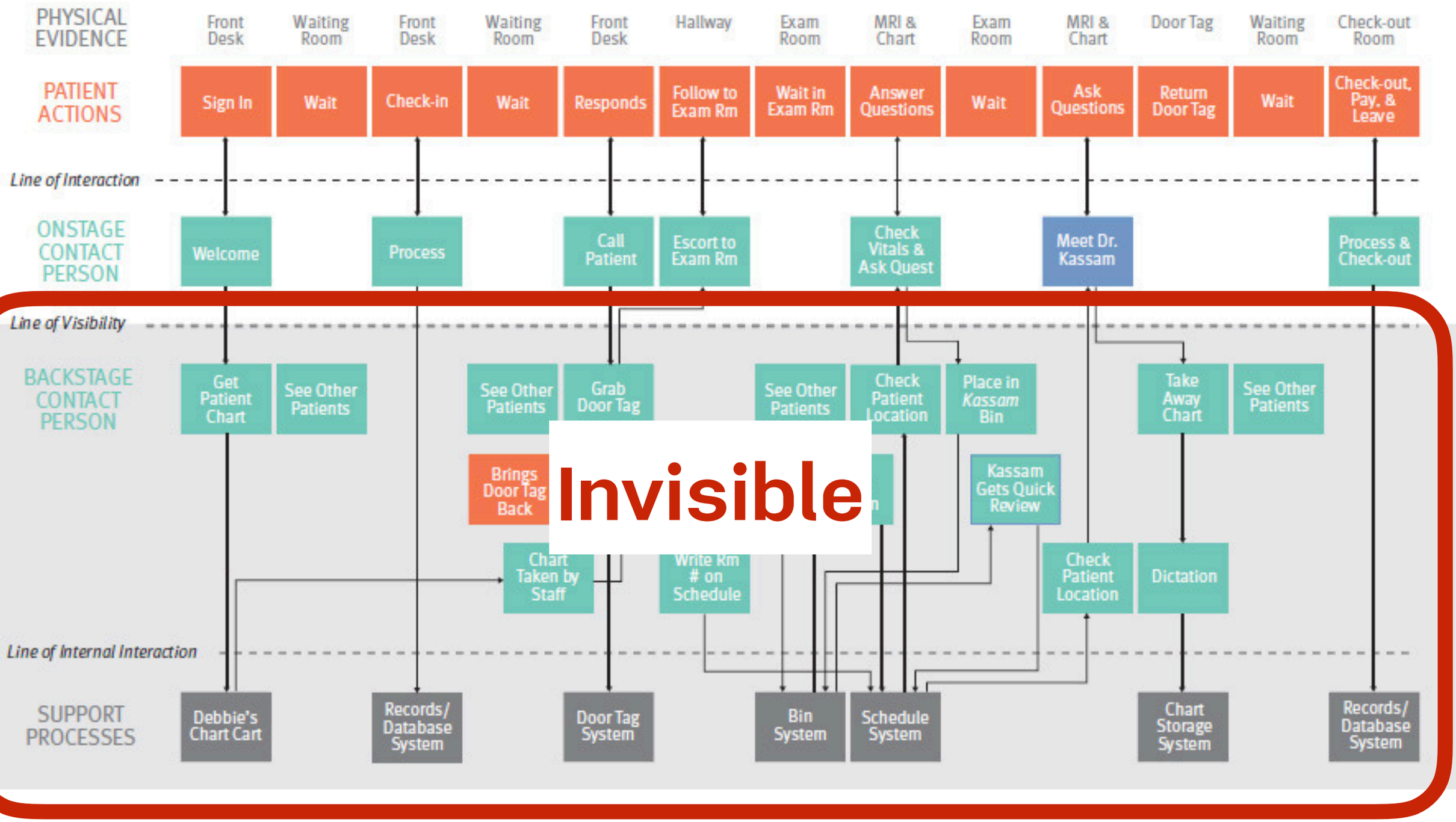
Service Blueprint of Presby Neuro Clinic



Service Blueprint of Presby Neuro Clinic



Service Blueprint of Presby Neuro Clinic



Invisible

Service Blueprint of Presby Neuro Clinic