Avancier Methods (AM)

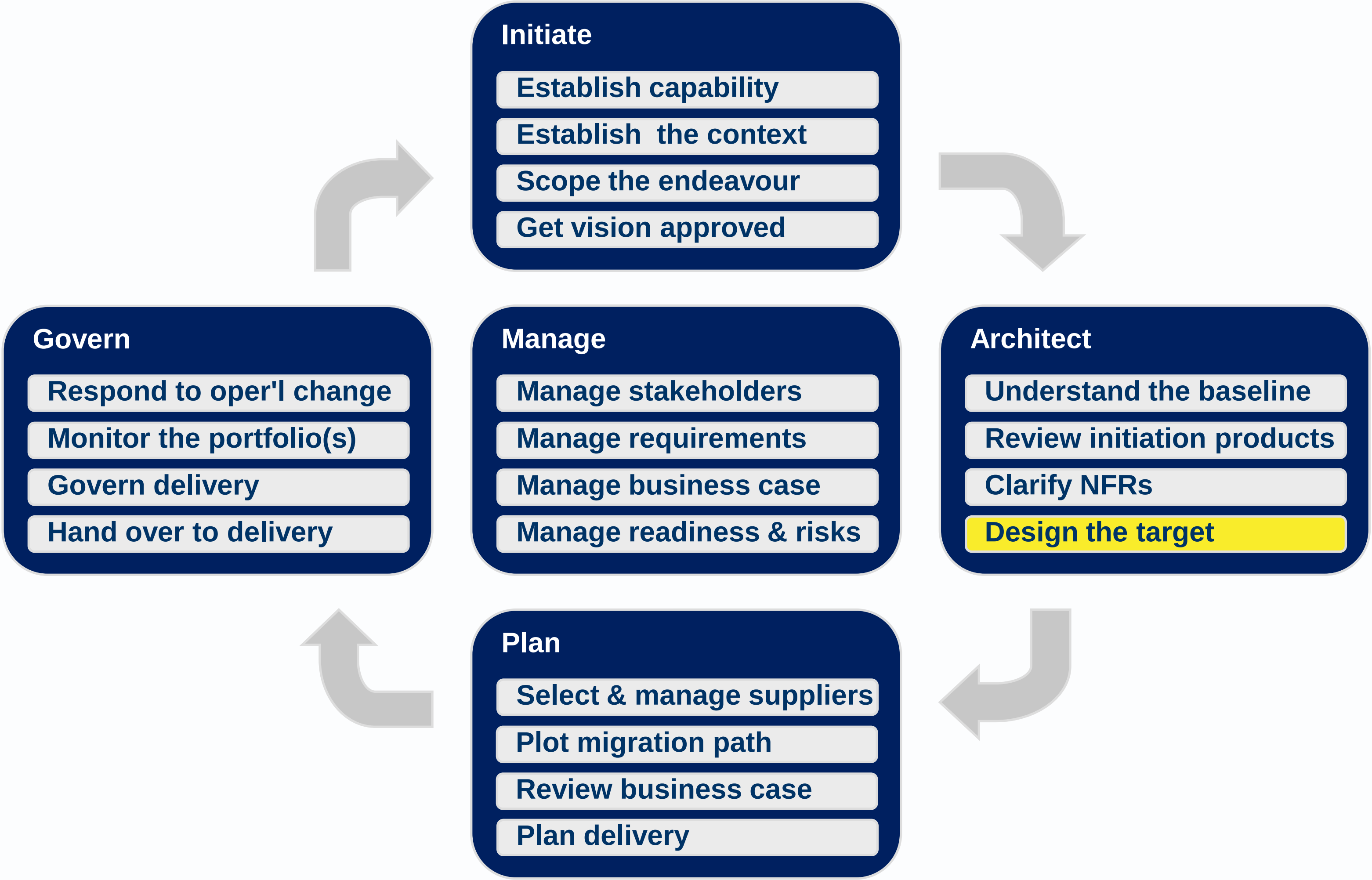
Solution Architecture

Design Platform Technology

CONTEXT

- What is the AM level 2 process?
- Which domain are we working in?
- What is the AM level 3 process?

AM level 2 process



Initiate
- Establish capability
- Establish the context
- Scope the endeavour
- Get vision approved

Govern
- Respond to oper'l change
- Monitor the portfolio(s)
- Govern delivery
- Hand over to delivery

Manage
- Manage stakeholders
- Manage requirements
- Manage business case
- Manage readiness & risks

Architect
- Understand the baseline
- Review initiation products
- Clarify NFRs
- Design the target

Plan
- Select & manage suppliers
- Plot migration path
- Review business case
- Plan delivery

Which domain are we working in?

	Passive Structure	*Required Behaviour*	*Logical Structure*	*Physical Structure*
Business		Business Service Business Process	Function Role	Org Unit Actor
Data / Information	Data Entity	Data Flow	Log Data Model	Data Store
Applications		IS Service	Application Interface	Application Component
Platform Technology		Technology Service	Technology Interface	Technology Component

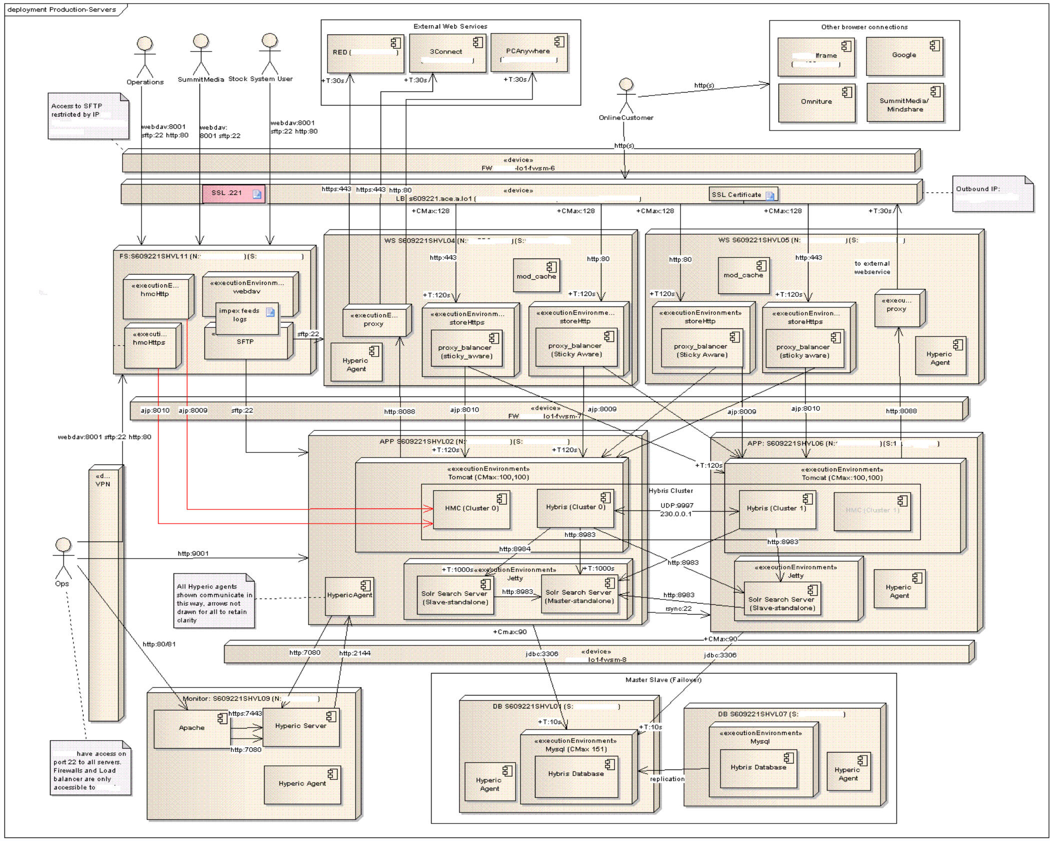
AM process level 3: Design Target Platform Technology

1. Identify requirements and context
2. Establish baseline opportunities and constraints
3. Define platform nodes
   - Clients
   - Data sources,
   - Others
4. Map software to platform nodes
5. Map logical nodes to physical nodes
6. Define the network
7. Refine to handle NFRs
8. Define non-production environments
9. Govern deployment and transition into operations

A process that takes you from a “logical” applications architecture to the definition of the platform technology infrastructure to support those applications

The end product?

- A diagram showing
  - deployment of software components to platform nodes
  - connections and protocols between components



Access to SFTP restricted by IP

All Hyperic agents shown communicate in this way, arrows not drawn for all to retain clarity

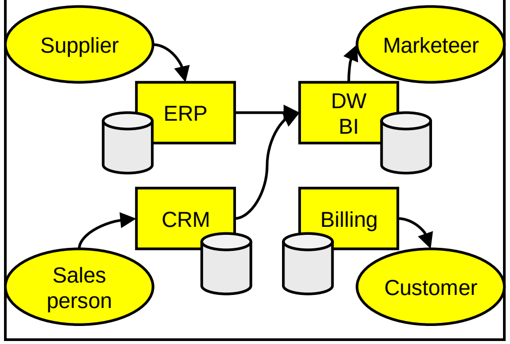
have access on port 22 to all servers. Firewalls and Load balancer are only accessible to

1 Identify requirements and context for IT

- First, collect what is known of

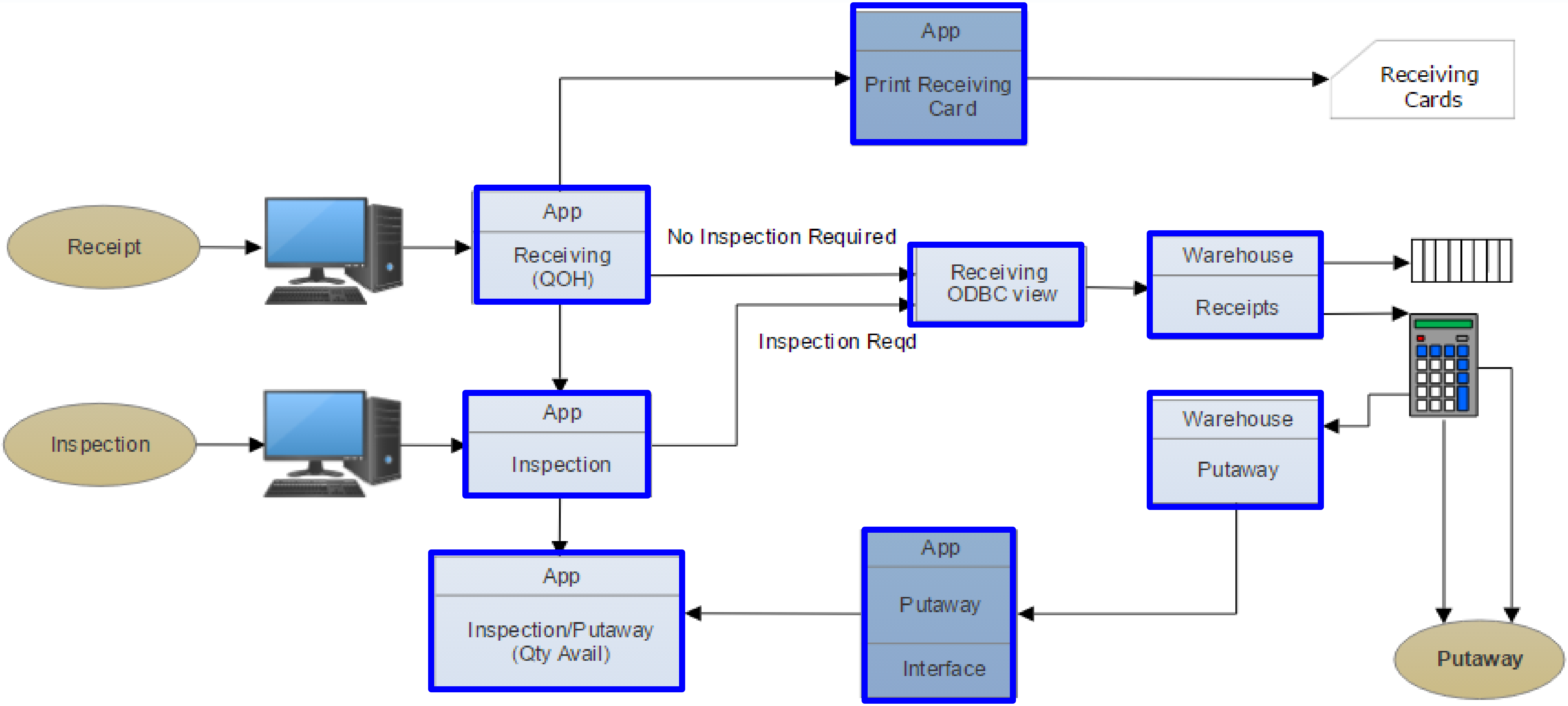
- Business applications
  - And the data flows or service invocations between them
- Geography
  - Locations of users, data stores and applications
- Non-functional requirements
  - Relating to users, data stores and applications
- Platform services
  - Needed by business applications

Data Flows between Subsystems
Applications Architecture Structure
Application Communication Diagram



Logical App Communication (aka Data Flow) Diagram

- No information about the application deployment to infrastructure



2 Establish baseline opportunity and constraints

- Does the enterprise constrain the choice and design of the infrastructure to support the target solution?
- By providing baseline infrastructure?
- By providing technology standards and directives?
- Can you use the existing infrastructure?

- Analyzing the current system configuration helps you understand what system performance depends on, the quality of the hardware, the configuration and the way the software works.
- You can use that information to tune and scale the overall architecture.

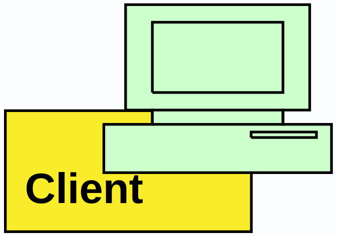
3 Define platform nodes (clients, then data sources, then others)

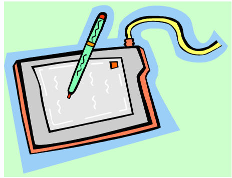
Show stoppers

- Define client-end devices
- Define data sources
- Define intermediate servers
  - (Label nodes with roles and identifiers)
- Define node operating systems

Define client-end devices

- Client devices can be a show stopper
- Where are they?
- What are they? Lap top specification must include any platform needed by the apps
- Who can and will use them? Employees must be willing and able to carry portable equipment
- How will they connect to servers?

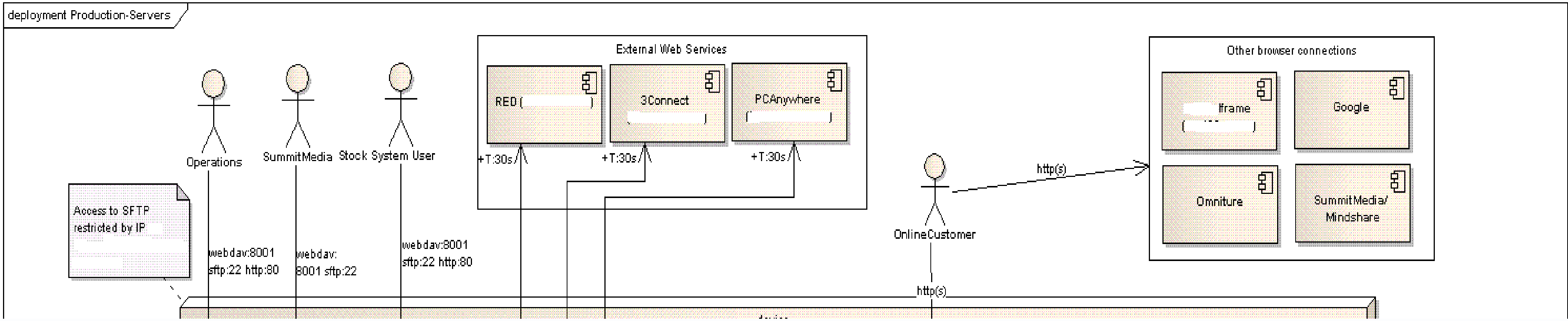




E.g. An ecommerce site

- Privileged users run some operational functions. Several protocols support use cases such as
  - upload files
  - set stock level to that in the warehouse
  - administer the site through the browser based admin interface.

- The public connect to the site through a browser (IE6+, Firefox, Chrome Safari, etc.)
- The http(s) connections to the right invoke services external to the solution, but important to the web-site (e.g. information used to track user journeys).
- We don’t see these on our servers.



- They access the file server and the admin interface.
- Secured by IP white listing and user name/password.

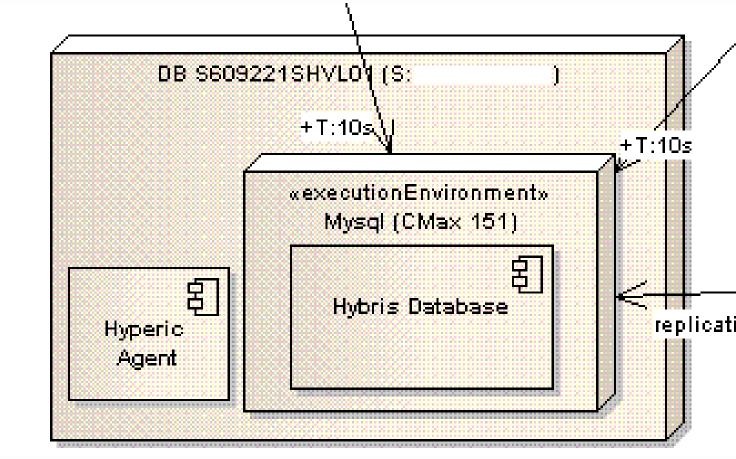
- They browse and use our application via the http(s) traffic going downwards

Define data sources

- The availability of data sources can be a show stopper

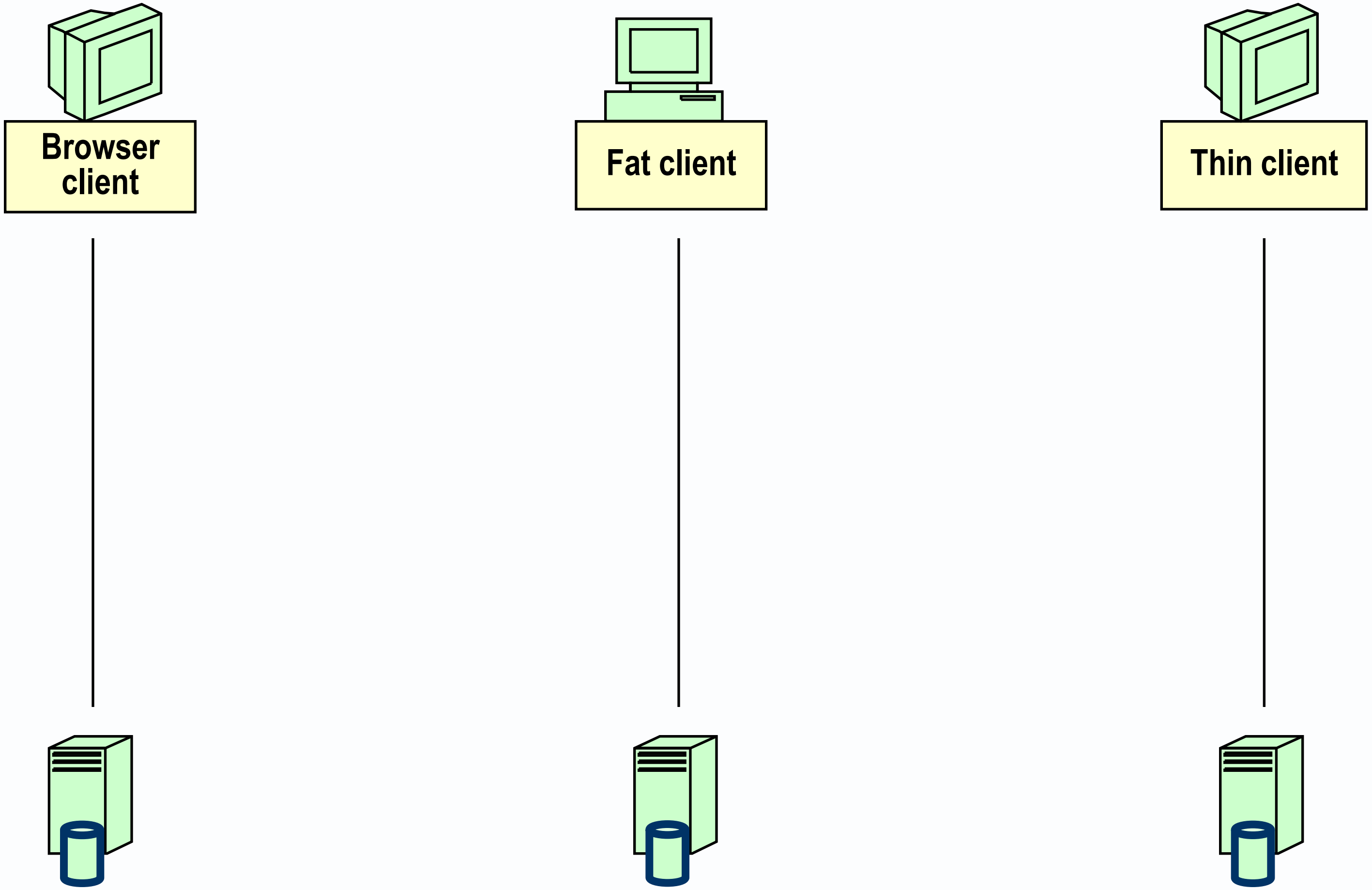
- Can existing data servers be used?
- Where is the data actually stored?
- Disc attached to data server? NAS? SAN?
- Is data available when needed - at right time of day?
- Are users authorised to access the data?

- In our example data is stored on a disc attached to the server.
- Data is available 24/7 (the website never closes) except for scheduled maintenance.



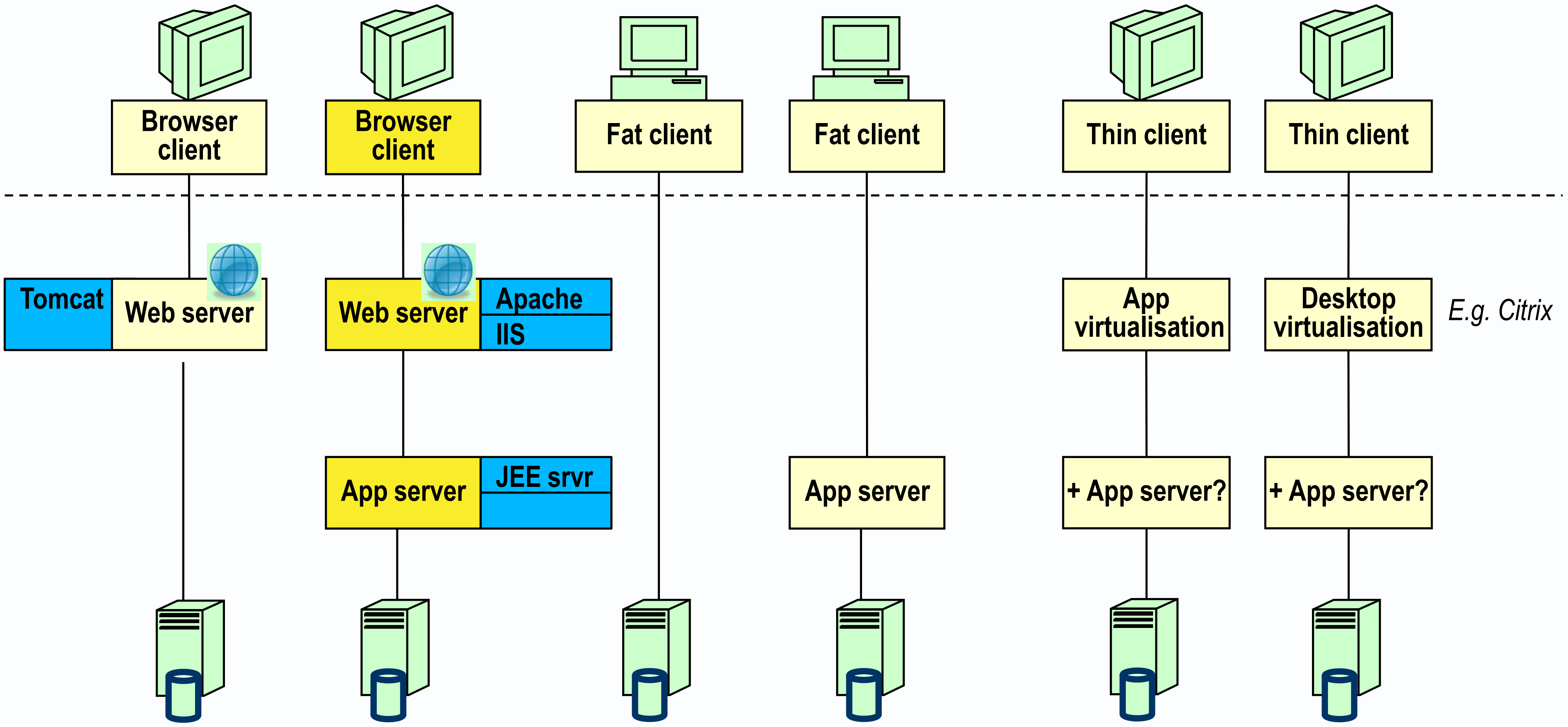
Define top and bottom nodes of client-server stack

- End-user device – Data source



Define intermediate nodes/tiers

- Each should have a defined role, be logically isolated from each other



Browser client

Browser client

Fat client

Fat client

Thin client

Thin client

Tomcat

Web server

Web server

Apache

IIS

App virtualisation

Desktop virtualisation

E.g. Citrix

App server

JEE srvr

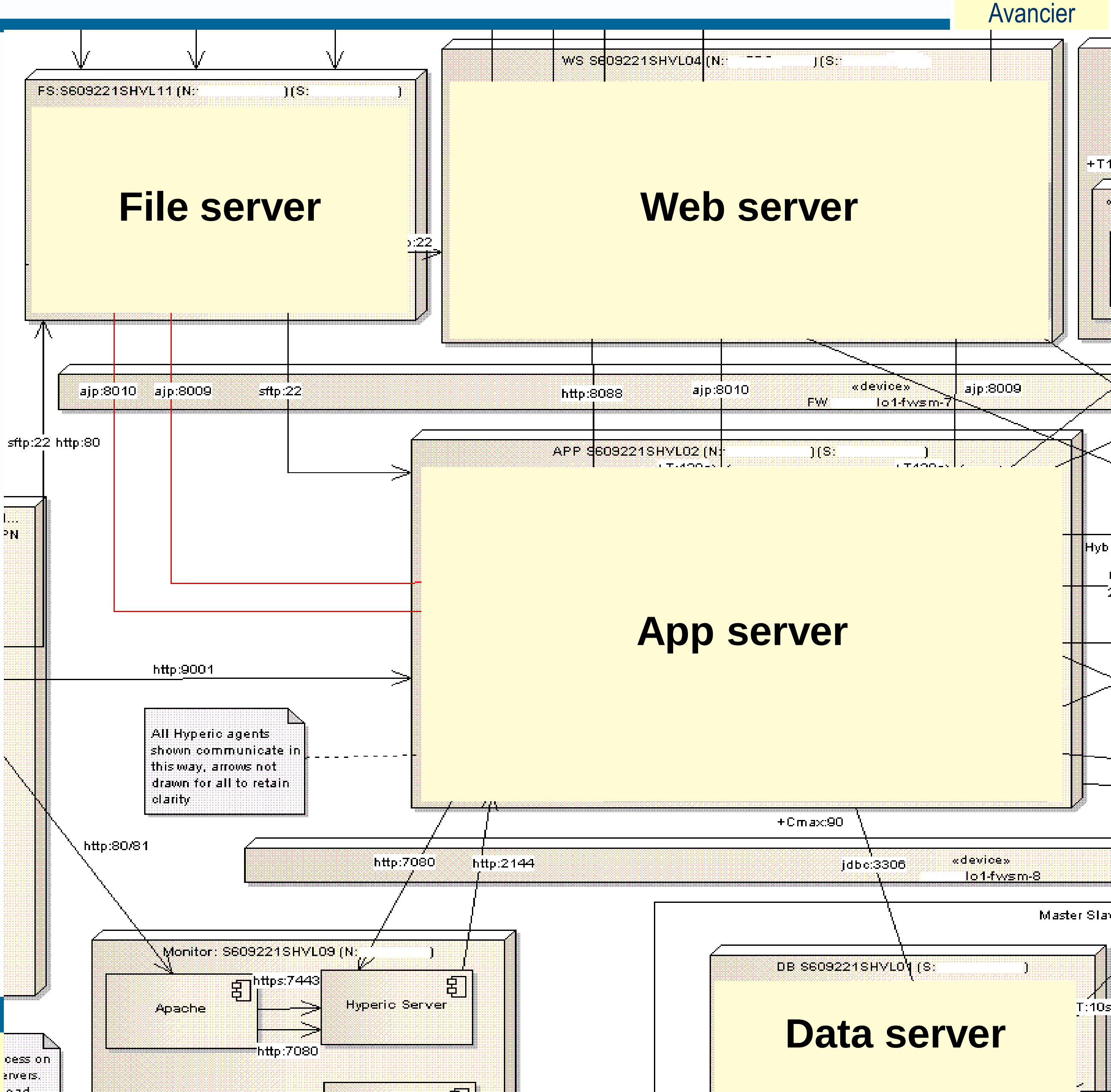
App server

+ App server?

+ App server?

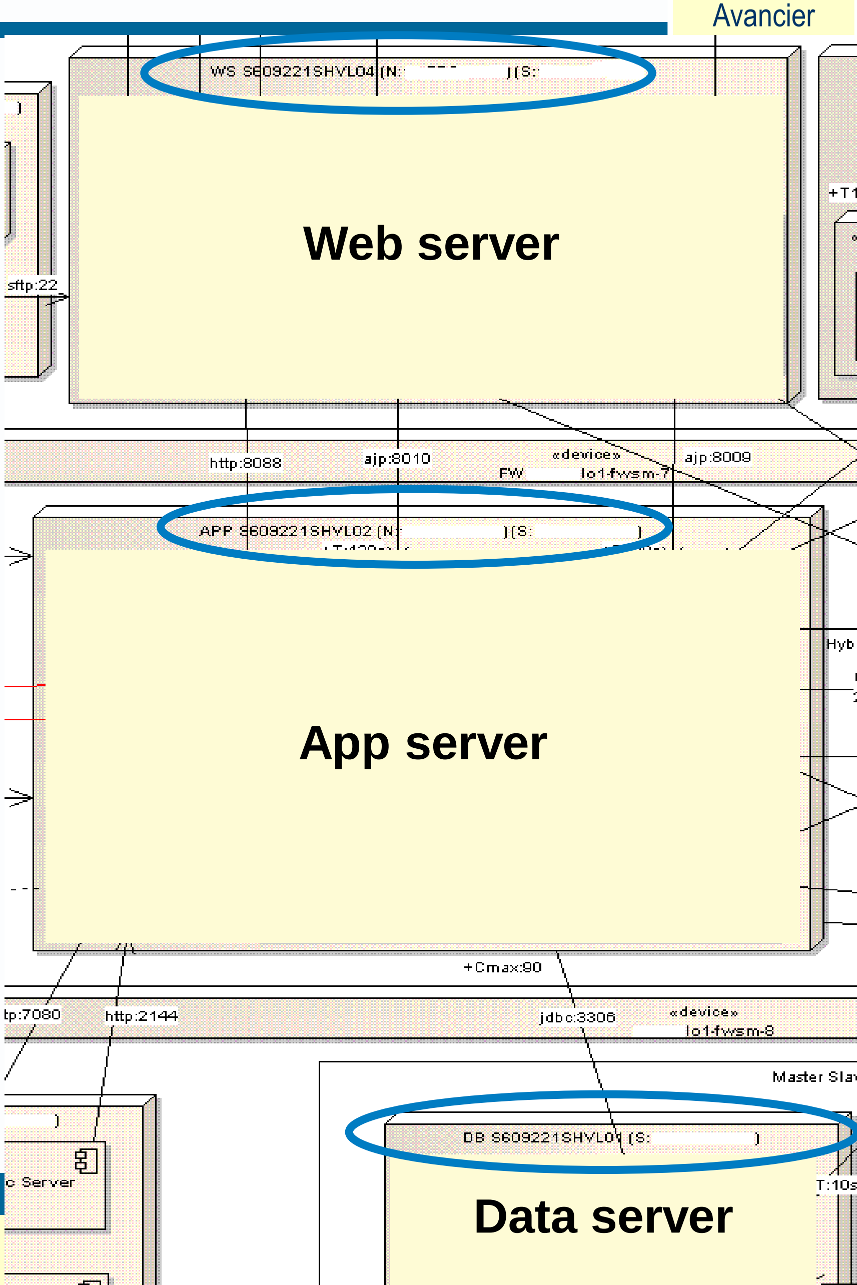
Define Intermediate Servers

- Platform nodes should
- have a defined role
- be logically isolated from each other
- occupy a specific tier


FS:S609221SHVL11 (N: [redacted])(S: [redacted])
File server
WS S609221SHVL04 (N: [redacted])(S: [redacted]
Web server
:22
ajp:8010
ajp:8009
sftp:22
http:8088
ajp:8010
«device»
FW [redacted] lo1-fwsm-7
ajp:8009
APP S609221SHVL02 (N: [redacted])(S: [redacted])
App server
sftp:22 http:80
http:9001
All Hyperic agents shown communicate in this way, arrows not drawn for all to retain clarity
http:80/81
+Cmax:90
http:7080
http:2144
jdbc:3306
«device»
[redacted] lo1-fwsm-8
Master Slav
Monitor: S609221SHVL09 (N: [redacted])
Apache
https:7443
Hyperic Server
http:7080
DB S609221SHVL01 (S: [redacted])
Data server
T:10s
+T1
Hyb

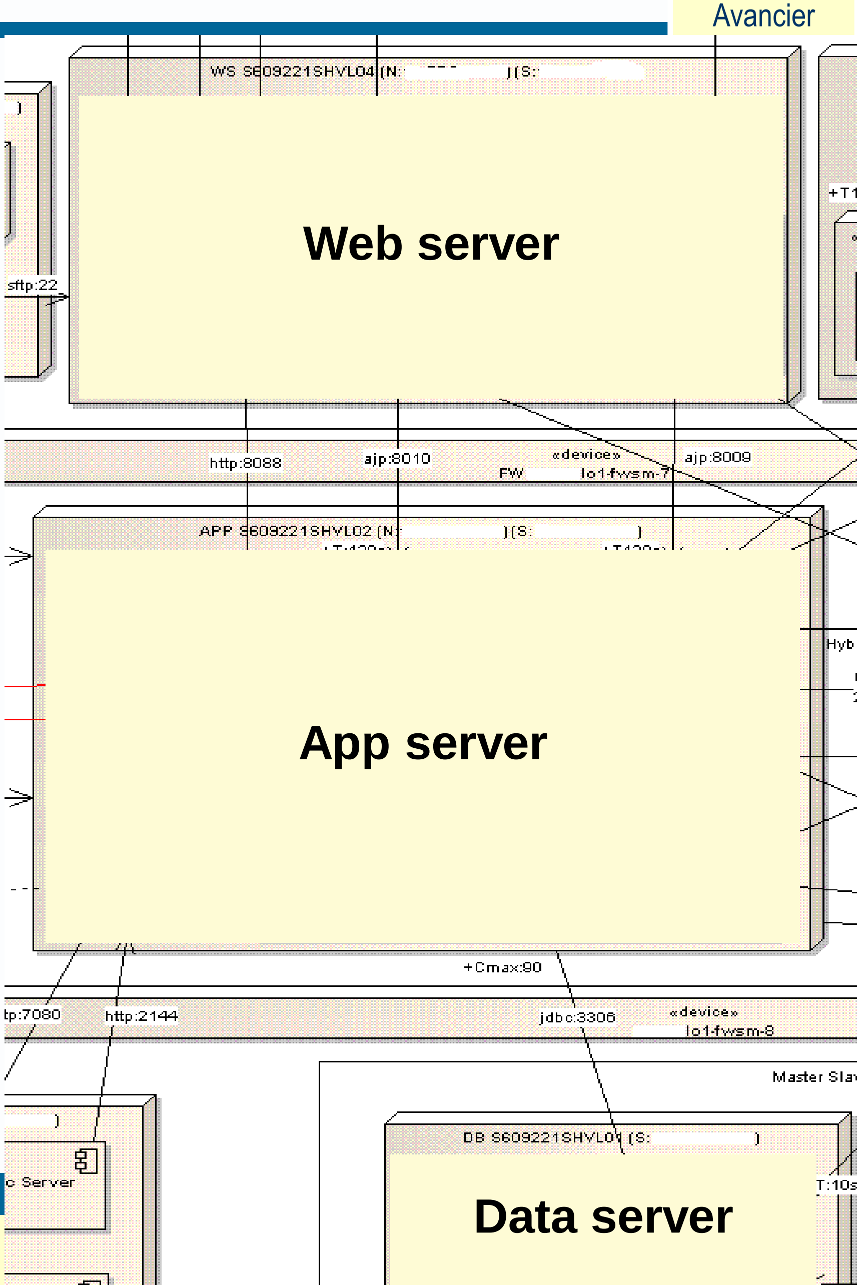
Label nodes with roles and identifiers

- Node Roles
  - Client = End user device
  - FS= File server,
  - WS = Web Server,
  - APP = App server,
  - DB = Database server
  - FW = Firewall
  - LB = Load balancer
  - Mainframe = MF
  - Monitor
- Node Identifiers
  - The server node shows the instance name of the server
  - The number has no meaning other than a unique identifier.



Define node operating systems

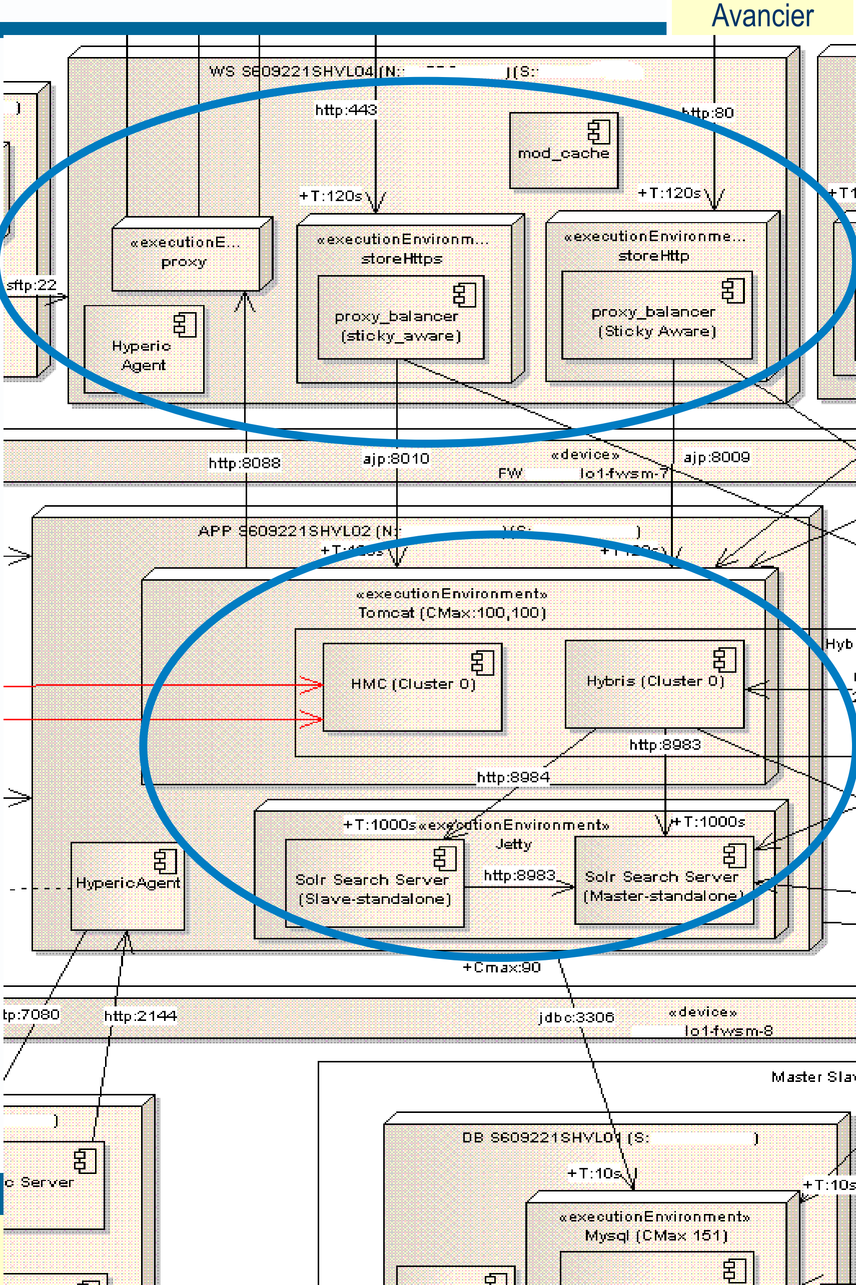
- In this example, all servers are Linux and on the same version/patch level,
- So no need to show on this diagram
- If you know one it is the same for all.



4 Map software to platform

- Assign each application software component to a platform node, or an execution environment within a platform node

- An execution environment is a type or part of a node that represents a particular execution platform, such as an operating system or a database management system.

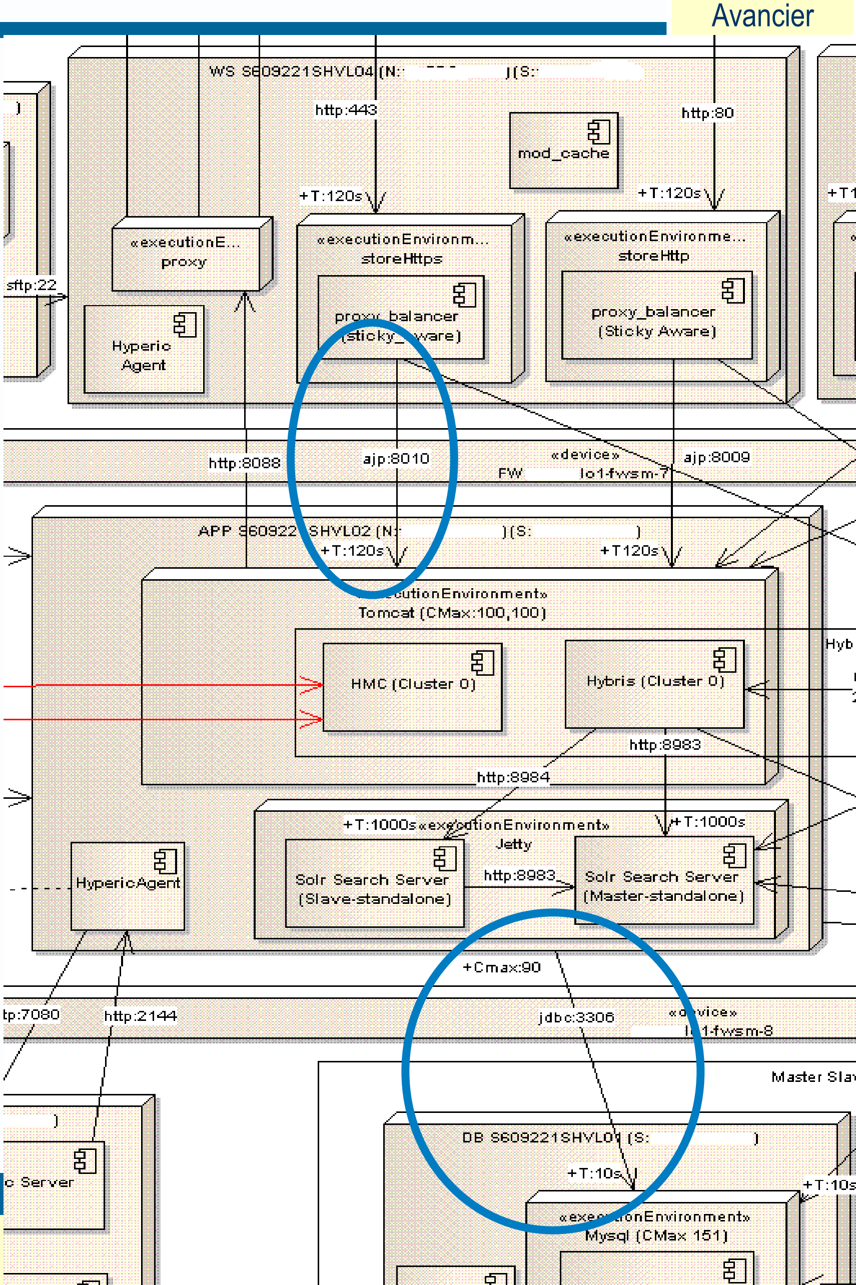


Define each inter-component connection

- Here, the arrow direction is
- From a component that initiates a connection

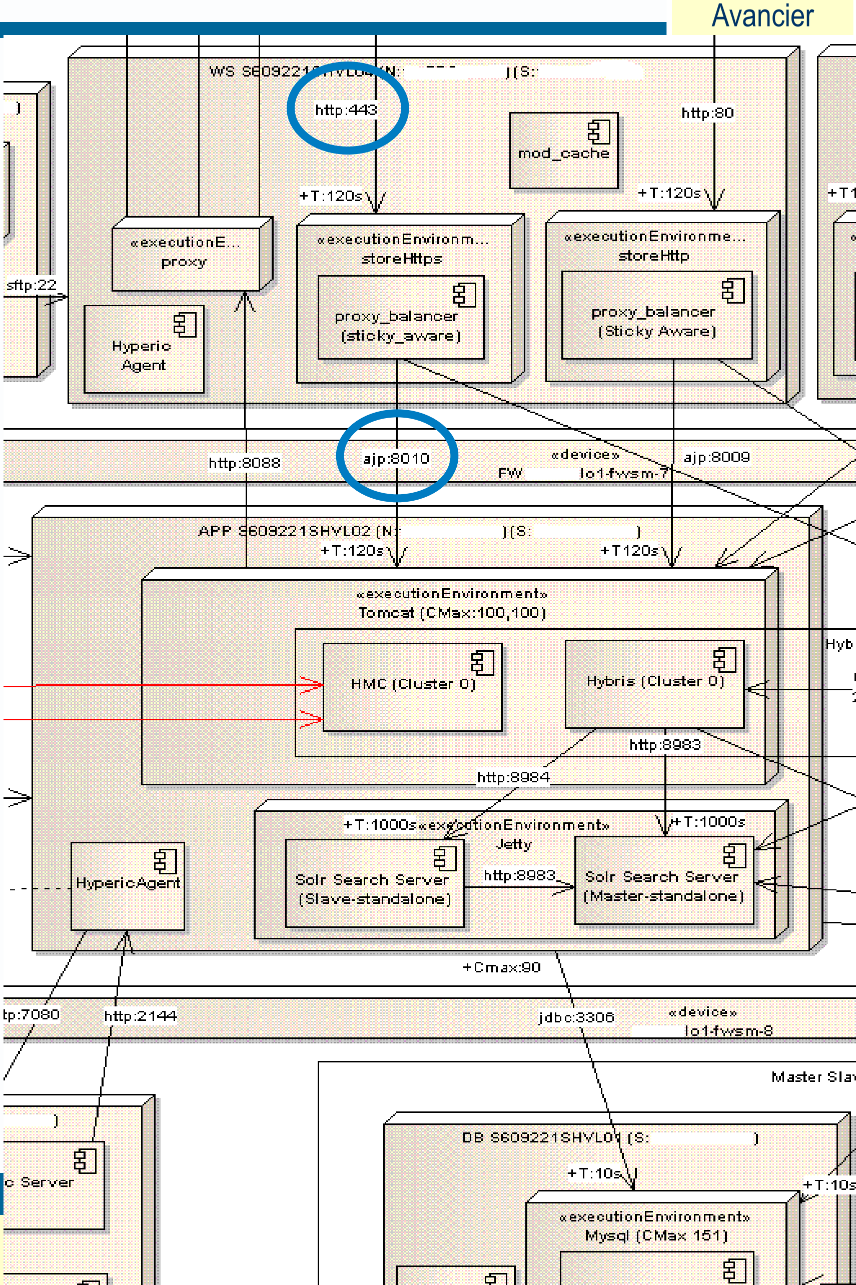
↓

- To the component that must be connected



Define the protocol used to make a connection

- Connections are generally annotated with an application layer protocol such as http or ajp.
- In this example, TCP is assumed, bar one instance of UDP which is shown later.
- (Most looking at this diagram are interested in the application layer protocols and do not separate protocol layers.)

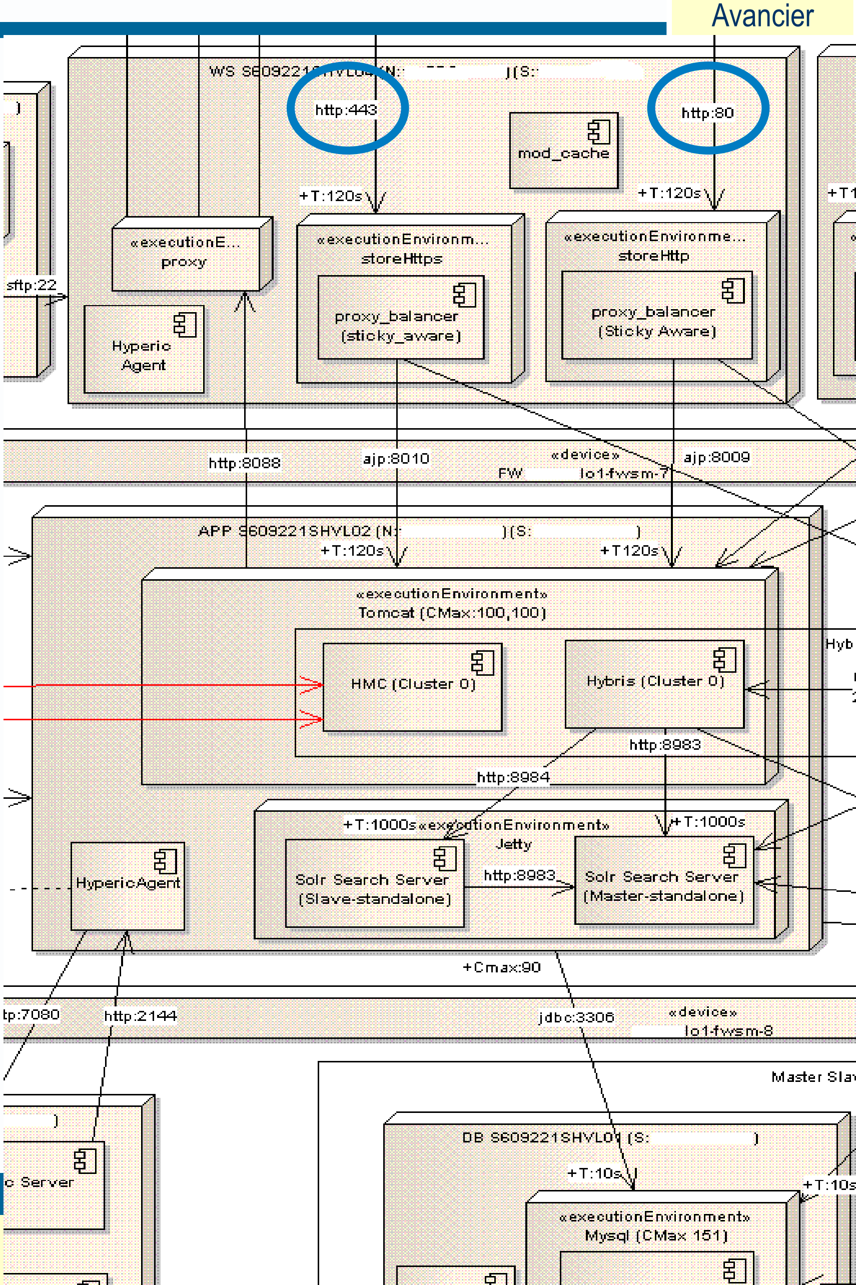


Define the port used for a connection

- The application layer protocol suggests the connection purpose and the ports it will likely be using.

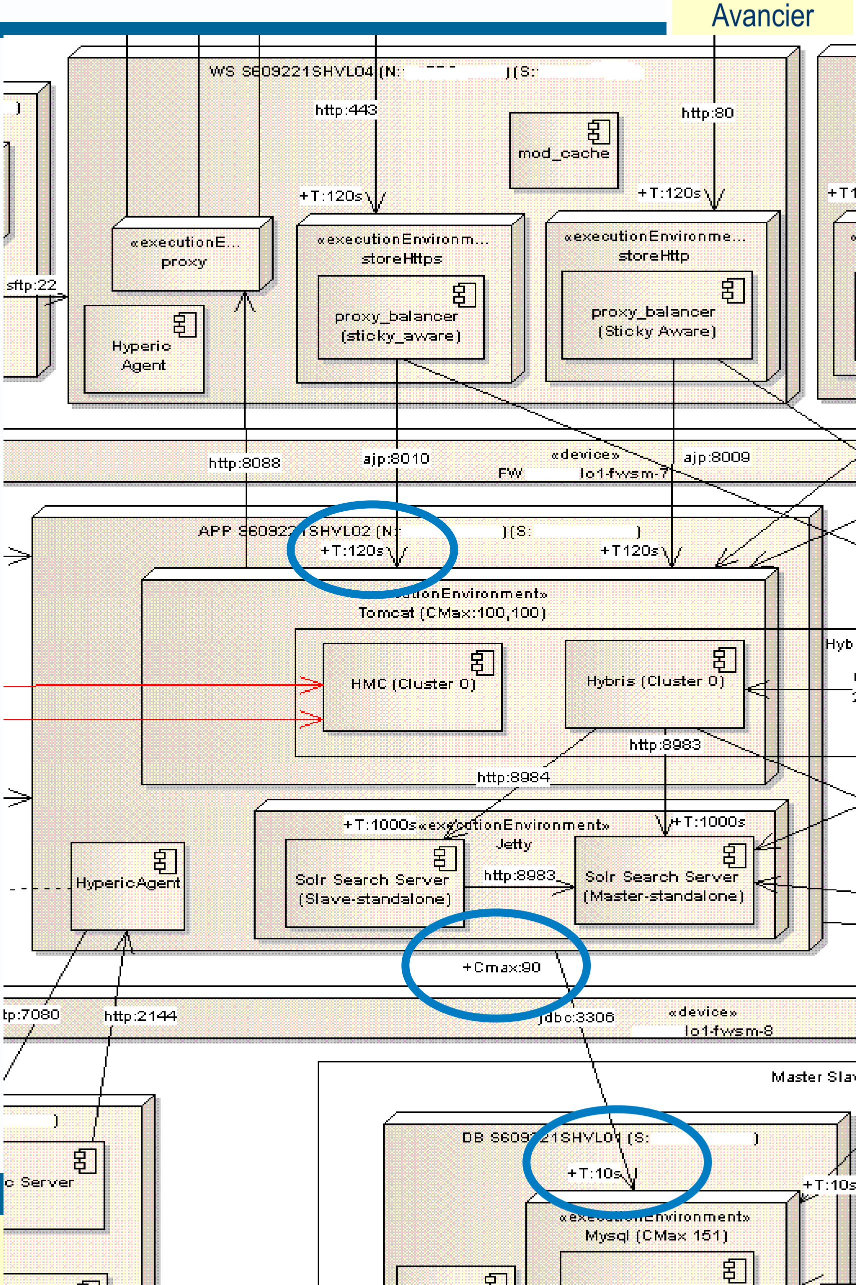
- However, showing ports helps the guys who need to manage the firewalls.

- (Why http:443? This traffic has been decrypted into plain text by the load balancer (hence the load balancer shows the SSL certificate). The 443 port is retained so that the lower layer knows that it was originally secured (encrypted) even though it is plain text now.)



Define physical characteristics

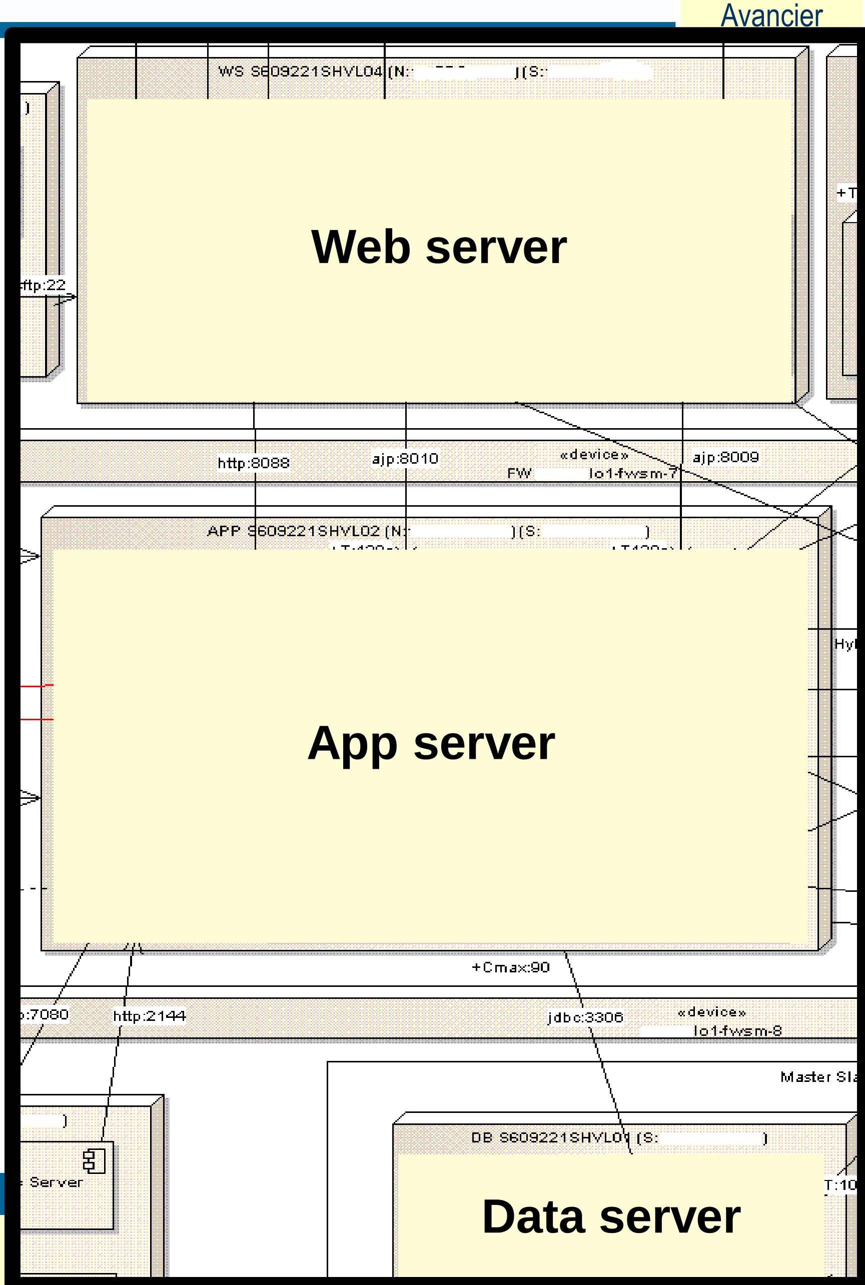
- Annotations can show additional information such as
- CMax (Maximum Connections)
- (Tout: connection time out)



5. Map logical nodes to physical nodes

- A physical node is a computer or other computing device such as a printer.
- Each has a unique network address, sometimes called a *Data Link Control (DLC) address* or *Media Access Control (MAC) address*.

- In this example, these are virtual server instances
- You can think of them as physical servers
- But they are **all deployed to a single (large) physical server.**



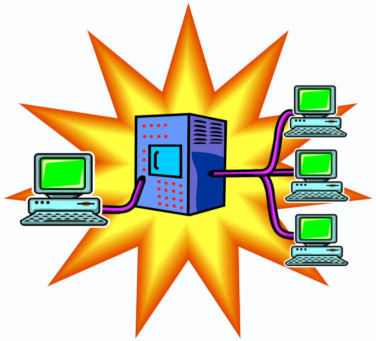
Size the physical node resources needed

- General sizing (2012) of processor and memory

Platform tier	Character	Processor CPU	Memory RAM
WEB Server	memory intensive	1	Large
APP server	processor intensive	4	Medium
DB server	memory and processor intensive	4	Large

6 Define the network

- Use data volume and frequency numbers to calculate the network bandwidth
- Consider network scope (PAN, LAN, MAN, WAN)
- Required network service layer (1, 2 or 3)
- Internal network devices (switches and routers)



- In our example, not shown, since the focus is on a single production server

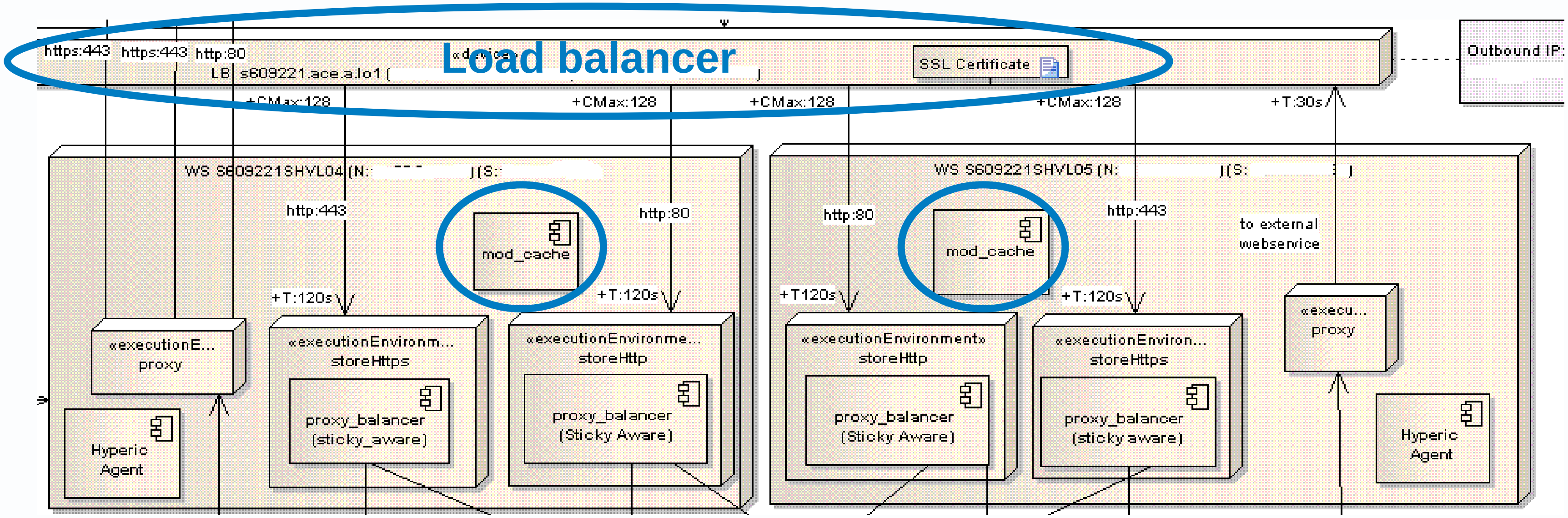
7 Refine to handle NFRs

- Iterate through the whole design to make sure all NFRs are considered and met

- Design for throughput and response time
- Design for availability
- Design for security (firewalls etc.)
- Design for serviceability
- Design for recoverability

WS: Design for throughput, availability and response time

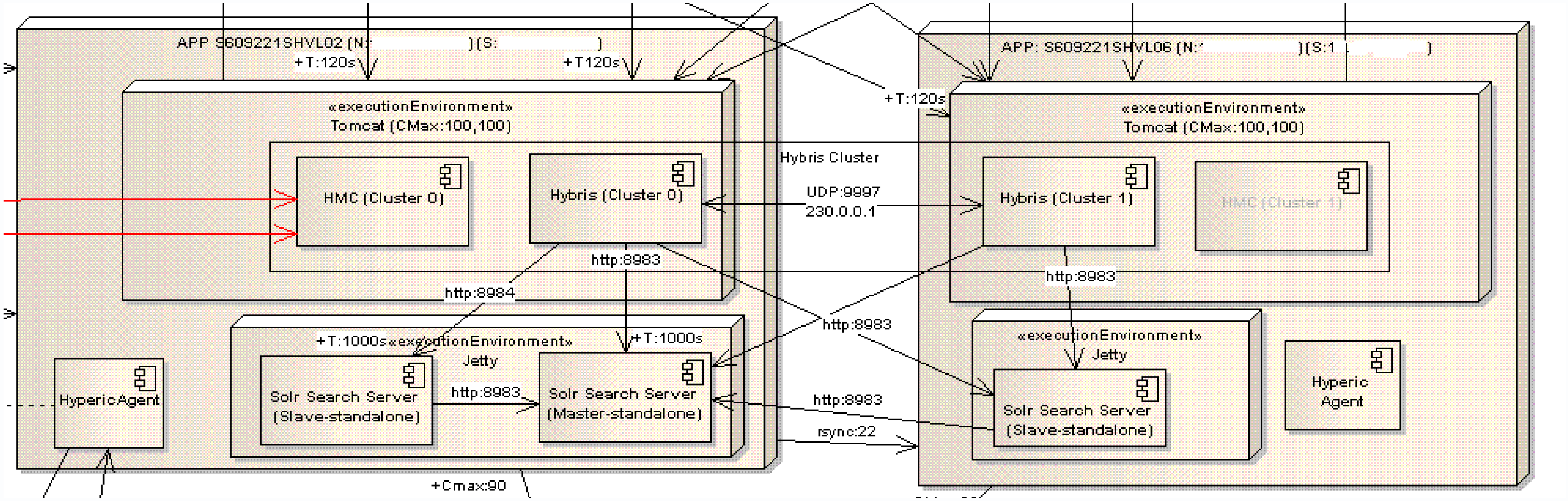
- A web site may have to handle thousands of users at once.
- First, a load balancer shares work between parallel web servers


Load balancer
https:443
https:443
http:80
«device»
LB | s609221.ace.a.lo1 ([redacted])
SSL Certificate
Outbound IP:
+CMax:128
+CMax:128
+CMax:128
+CMax:128
+T:30s
WS S609221SHVL04 (N: [redacted])(S: [redacted])
http:443
mod_cache
http:80
+T:120s
+T:120s
«executionE... proxy
«executionEnvironm... storeHttps
proxy_balancer (sticky_aware)
«executionEnvironme... storeHttp
proxy_balancer (Sticky Aware)
Hyperic Agent
WS S609221SHVL05 (N: [redacted])(S: [redacted])
http:80
mod_cache
http:443
to external webservice
+T120s
+T:120s
«executionEnvironment» storeHttp
proxy_balancer (Sticky Aware)
«executionEnviron... storeHttps
proxy_balancer (sticky aware)
«execu... proxy
Hyperic Agent

- Then, the "mod_cache" is important for performance
- Make the web-tier do as much as it can e.g. serve static content quickly.
- Having spent ages getting the HTML and images for a given URL from the app tier, cache it as static information, so you can serve a request for the same URL much faster.
- The web tier cache is very fast in comparison to making the app do some processing
- Without the cache you might need 20 app servers to serve the same number of users.

APP: Design for throughput and availability

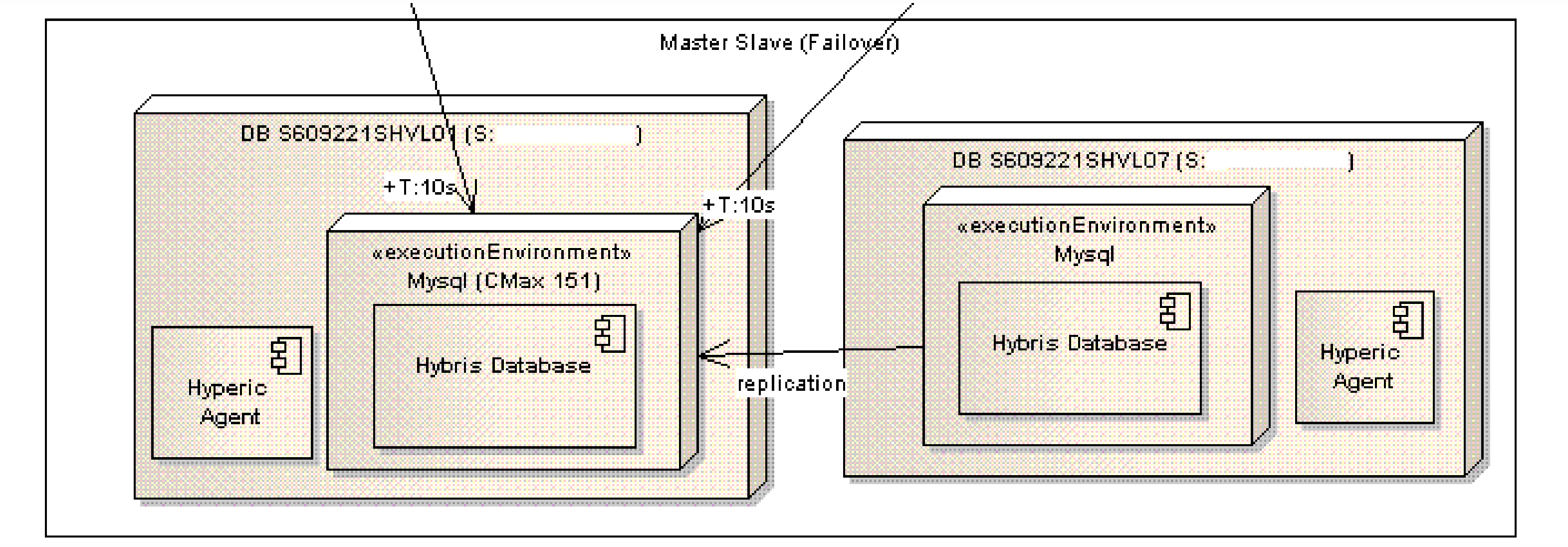
- Parallel app servers - configured in a cluster for resilience.



- The UDP connection synchronises cached objects on each server.
- No problem if the occasional update is missed
- The broadcast feature of UDP means that scaling out can happen easier, additional servers brought online without explicit re-configuration of the existing server.

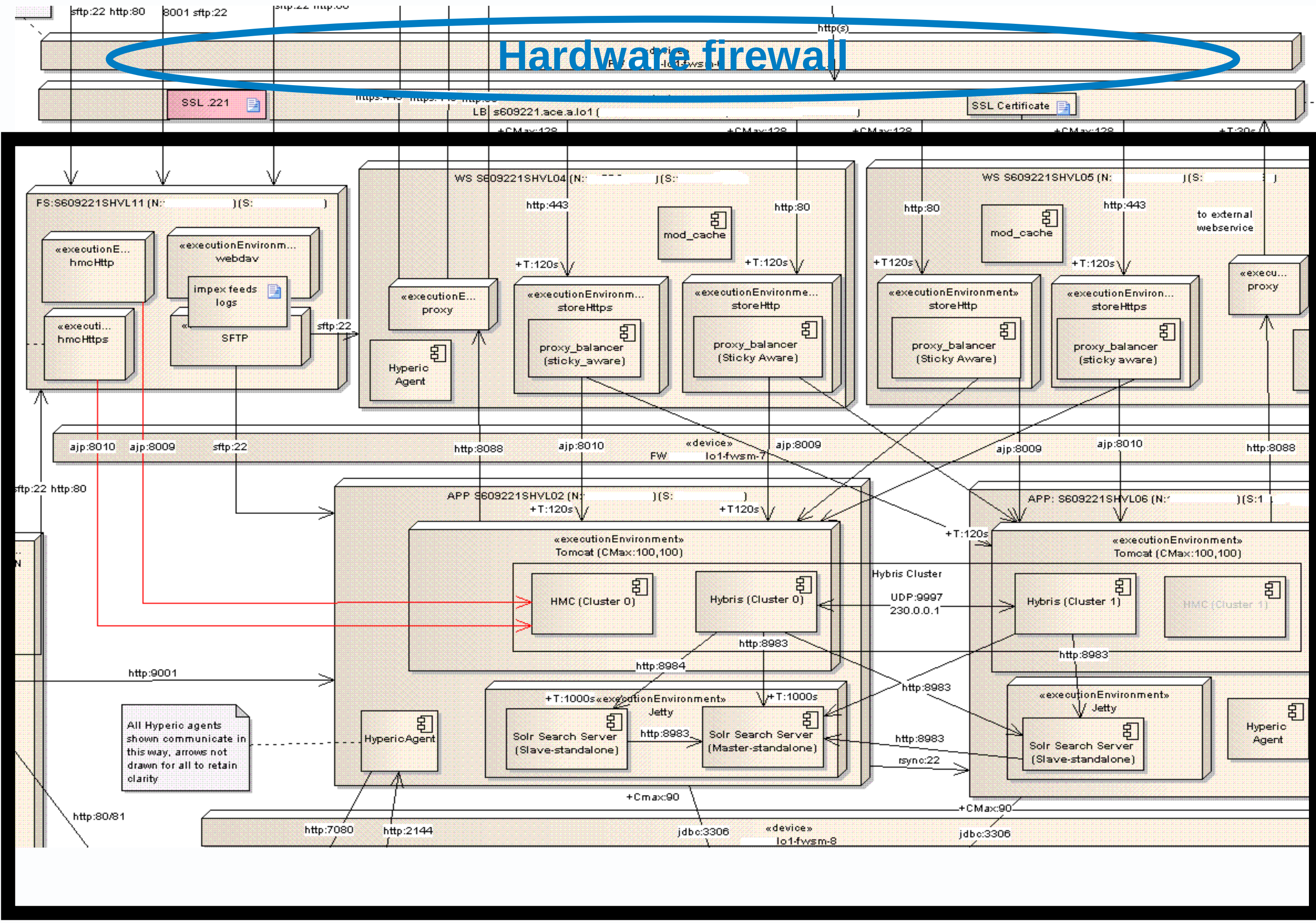
DB: Design for availability

- E.g. active-passive dual data server
- Dual data servers in a master-slave configuration
- The wider system only talks to the master.
- The slave makes regular copies of the data in the master.
- In the event of a failure of the master the system can be re-configured quickly to point to the slave which will then assume the role of master.



Design for security (hardware firewall)

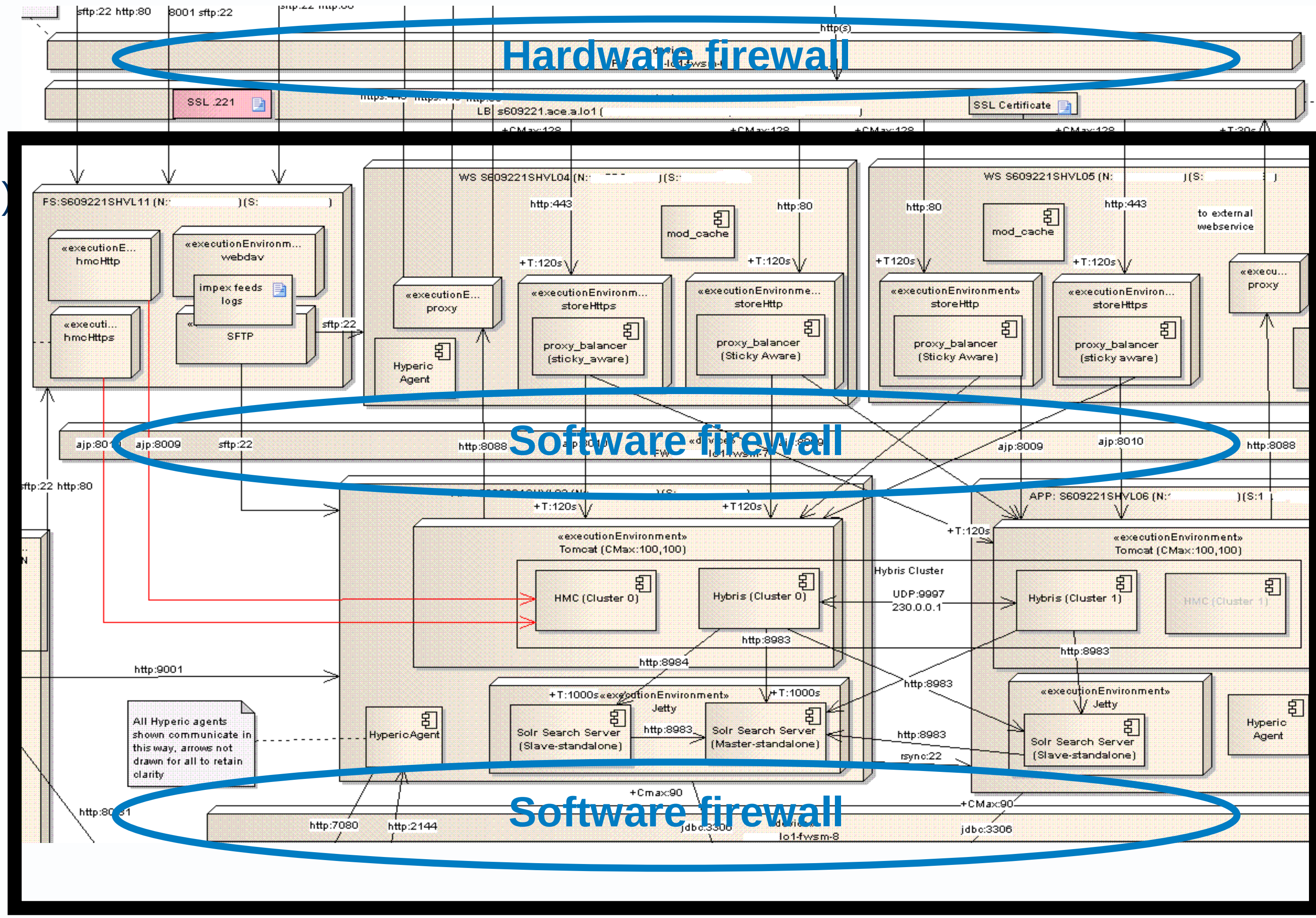
- The hardware firewall can decrypt the input data flow much faster than software
- It does mean a message is decrypted (from HTTPS to HTTP) before it enters the DMZ
- So, an Intrusion Protection System (IPS) may be needed as well



Design for security (software firewalls)

- Firewall separate adjacent tiers
- They implement strict packet filtering for both
- inward connections (ingres) and
- outward connections (egres)

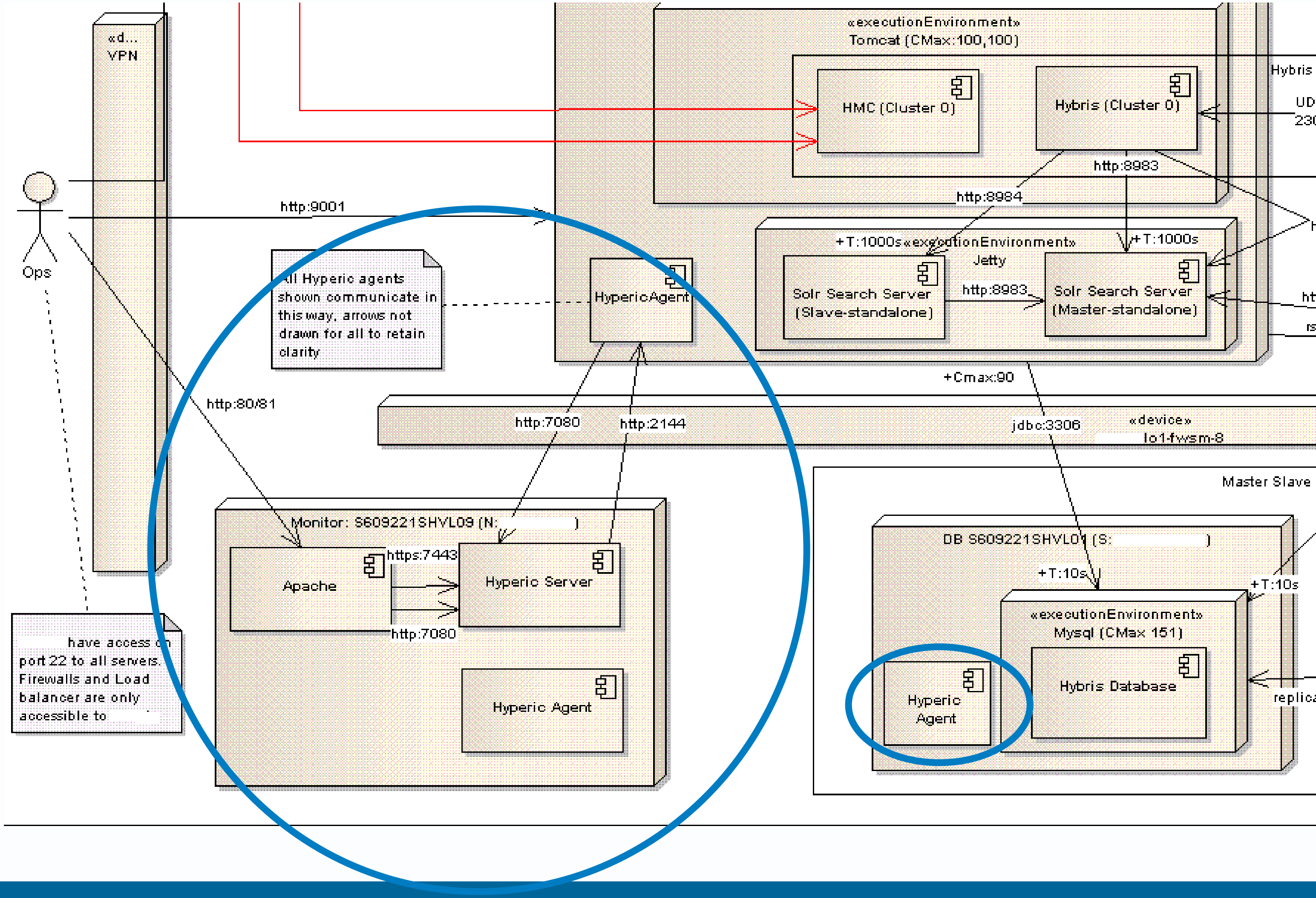
- This ensures communication between components of a whole application can only happen on the ports designated for that application.
- E.g. if a hacker managed to get onto the database server, they cannot establish a connection to the internet.



Avancier

Design for serviceability

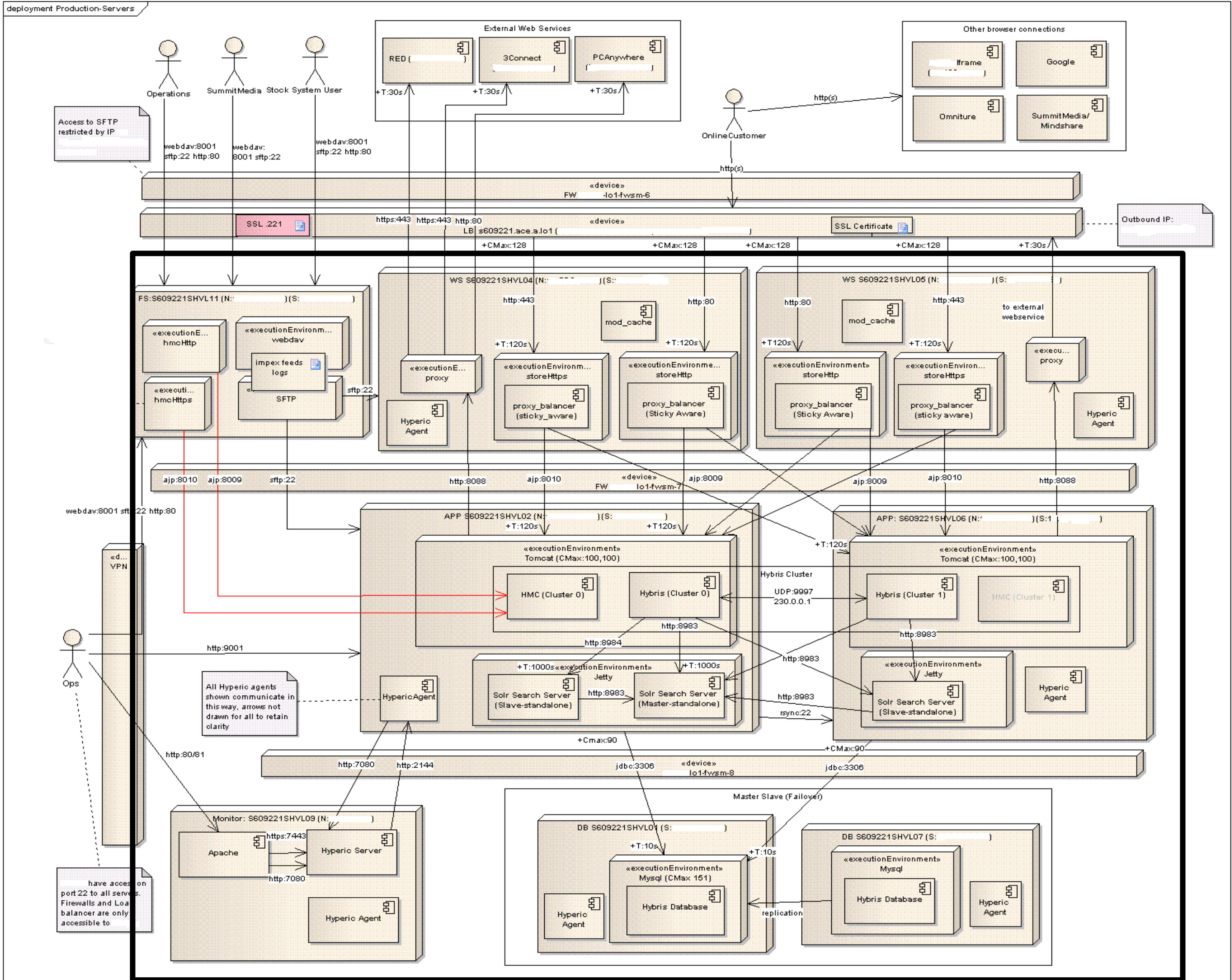
- The monitor tracks resource utilisation of all virtual servers.
- Might spin up a new server if needed?



All Hyperic agents shown communicate in this way, arrows not drawn for all to retain clarity

have access on port 22 to all servers. Firewalls and Load balancer are only accessible to

The result


deployment Production-Servers
External Web Services
RED ([redacted])
3Connect
[redacted]
PCAnywhere
[redacted]
+T:30s
+T:30s
+T:30s
Other browser connections
Iframe
[redacted]
Google
Omniture
SummitMedia/
Mindshare
Operations
SummitMedia
Stock System User
OnlineCustomer
http(s)
http(s)
Access to SFTP
restricted by IP
[redacted]
webdav:8001
sftp:22 http:80
webdav:
8001 sftp:22
webdav:8001
sftp:22 http:80
«device»
FW [redacted]-lo1-fwsm-6
SSL .221
https:443
https:443
http:80
«device»
LB s609221.ace.a.lo1 ([redacted])
SSL Certificate
Outbound IP:
[redacted]
+CMax:128
+CMax:128
+CMax:128
+CMax:128
+T:30s
FS:S609221SHVL11 (N:[redacted]) (S:[redacted])
«executionE...
hmcHttp
«executionEnvironm...
webdav
impex feeds
logs
«executi...
hmcHttps
SFTP
sftp:22
WS S609221SHVL04 (N:[redacted]) (S:[redacted])
http:443
http:80
mod_cache
+T:120s
+T:120s
«executionE...
proxy
«executionEnvironm...
storeHttps
«executionEnvironme...
storeHttp
proxy_balancer
(sticky_aware)
proxy_balancer
(Sticky Aware)
Hyperic
Agent
WS S609221SHVL05 (N:[redacted]) (S:[redacted])
http:80
http:443
mod_cache
to external
webservice
+T:120s
+T:120s
«executionEnvironment»
storeHttp
«executionEnviron...
storeHttps
proxy_balancer
(Sticky Aware)
proxy_balancer
(sticky aware)
«execu...
proxy
Hyperic
Agent
ajp:8010
ajp:8009
sftp:22
http:8088
ajp:8010
«device»
FW [redacted] lo1-fwsm-7
ajp:8009
ajp:8009
ajp:8010
http:8088
webdav:8001 sftp:22 http:80
«d...
VPN
APP S609221SHVL02 (N:[redacted]) (S:[redacted])
+T:120s
+T120s
«executionEnvironment»
Tomcat (CMax:100,100)
HMC (Cluster 0)
Hybris (Cluster 0)
Hybris Cluster
UDP:9997
230.0.0.1
APP: S609221SHVL06 (N:[redacted]) (S:[redacted])
+T:120s
«executionEnvironment»
Tomcat (CMax:100,100)
Hybris (Cluster 1)
HMC (Cluster 1)
http:8983
http:8984
http:8983
http:8983
Ops
http:9001
+T:1000s
«executionEnvironment»
Jetty
+T:1000s
Solr Search Server
(Slave-standalone)
http:8983
Solr Search Server
(Master-standalone)
http:8983
rsync:22
«executionEnvironment»
Jetty
Solr Search Server
(Slave-standalone)
Hyperic
Agent
All Hyperic agents
shown communicate in
this way, arrows not
drawn for all to retain
clarity
HypericAgent
http:80/81
+Cmax:90
+CMax:90
http:7080
http:2144
jdbc:3306
«device»
[redacted] lo1-fwsm-8
jdbc:3306
Master Slave (Failover)
Monitor: S609221SHVL09 (N:[redacted])
Apache
https:7443
Hyperic Server
http:7080
Hyperic Agent
DB S609221SHVL01 (S:[redacted])
+T:10s
+T:10s
«executionEnvironment»
Mysql (CMax 151)
Hybris Database
Hyperic
Agent
replication
DB S609221SHVL07 (S:[redacted])
«executionEnvironment»
Mysql
Hybris Database
Hyperic
Agent
[redacted] have access on
port 22 to all servers.
Firewalls and Load
balancer are only
accessible to [redacted]

Design for recoverability

- Duplicate resources at a remote disaster recovery site
- The entire configuration is replicated.
- If all important state information is stored in the database, then only that tier needs regular replication across data centres.
- Also, all servers should be "backed up" off-site on a daily basis.

8 Define non-production environments

- Remember s/w licence costs can rise with each real CPU and each virtual machine or LPAR

Environment type	Purpose: To	Physical platform	Hosted at Location	Contains Application Components	Contains Technology Components
Prototyping	Test/demonstrate a specific technology or design concept	1			
Development	Enable developers to write code	1			
System test	Enable system testers to the product	1			
Integration test	Test how the system integrates with others	2			
Performance test	Test how the system performs when fully loaded	2			
Data migration	Enable cleansing and migration of data	3			
User acceptance test	Enable user representatives to test to the product	4			
Production	Enable live operation of the system(s)	4			
Production support	Enable fault replication and investigation, and minor changes	5			

9 Govern deployment and transition into operations

- Monitor the progress of the deployment and refine the infrastructure design as need be

FOOTNOTES

Solution technology definition

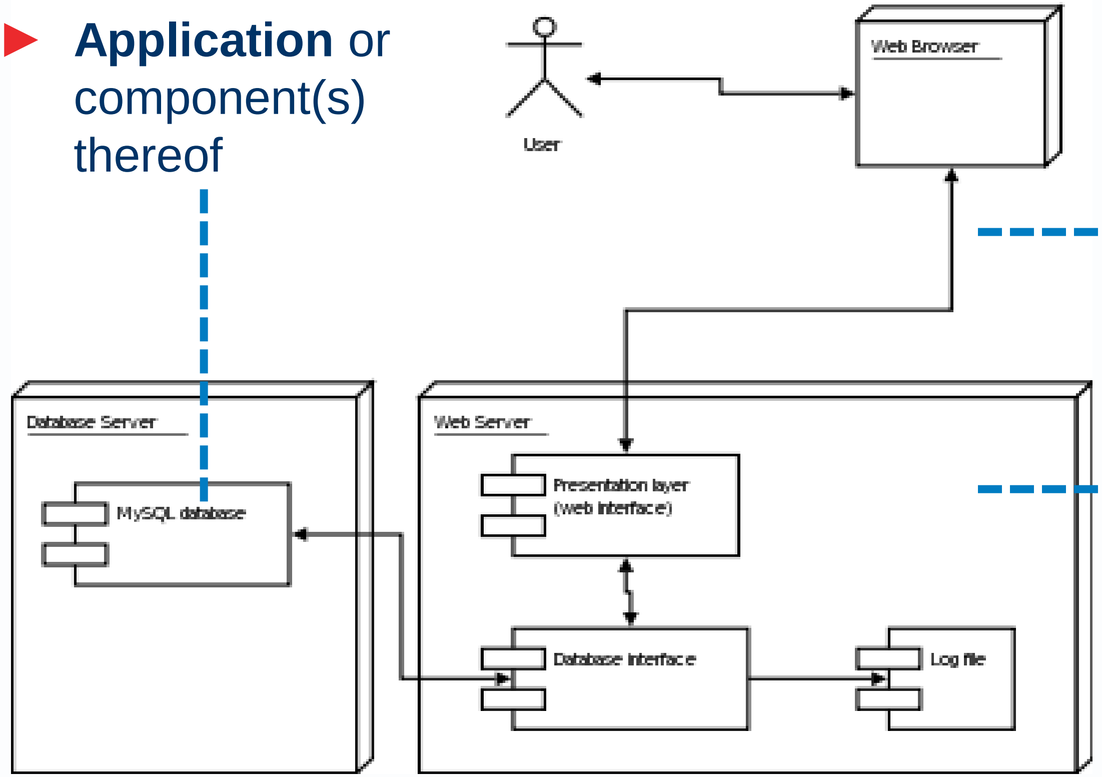
A process that progresses through stages from

- a logical application-information view, through
- progressively more physical views up to
- a hardware configuration diagram.

A process for defining the technologies that will support and run an application

UML deployment diagram

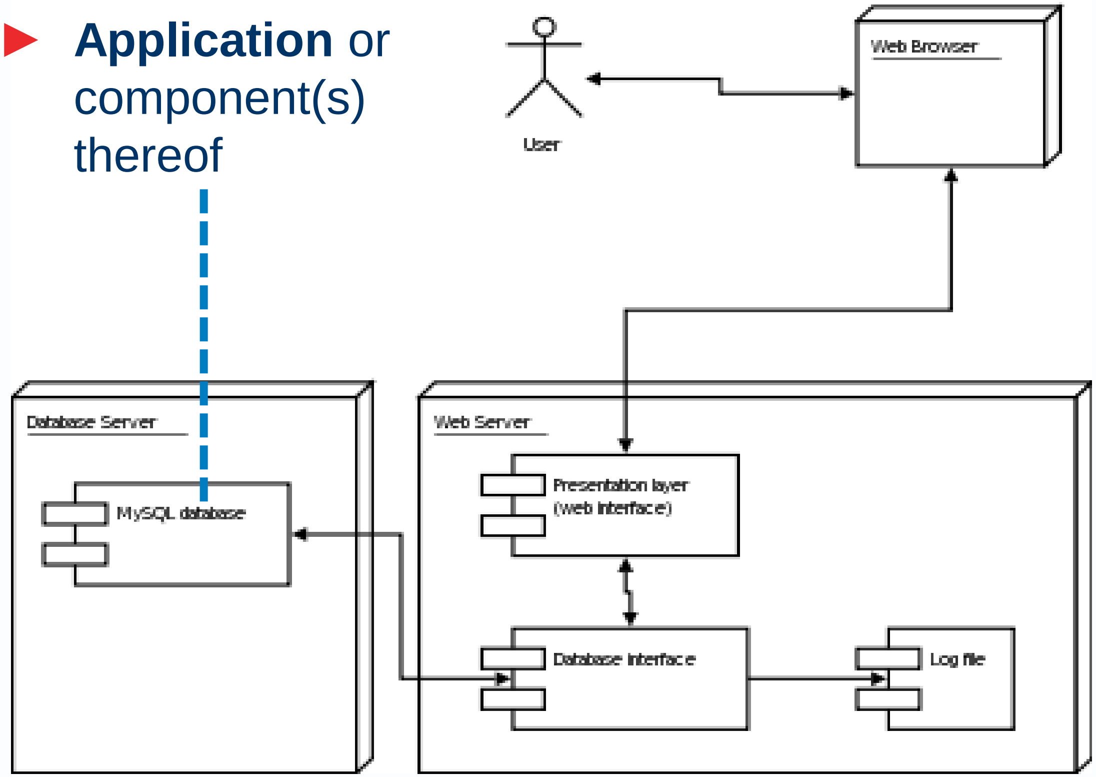
- **Application** or component(s) thereof



- **Communication path :** an association between two nodes, through which they are able to exchange signals and messages.
- **Node:** A computational resource upon which artifacts may be deployed for execution.
- Nodes can be interconnected through communication paths to define a network structure or topology.
- Nodes can be virtual or physical servers.

Deployment

- **Application** or component(s) thereof

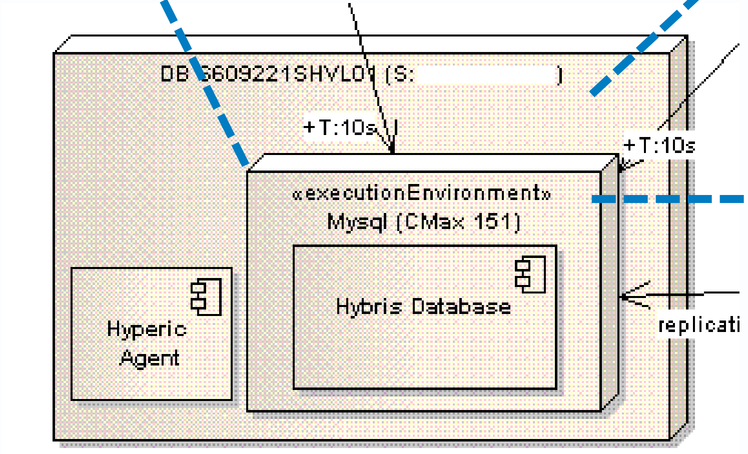


- An application is deployed by publishing its components on a node where platform technologies can read and execute the processes of those components.

- A deployable artefact packages some software components for deployment to one node of a computing network.
- It contains not only software components, but also meta data – which describes the software components.

Two kinds of node

Execution environment instances are assigned to **device** instances.



- **Device:** a node with memory and processing capability upon which artifacts may be deployed for execution.
  - E.g. application server, client workstation, mobile device, embedded device.
- **Execution environment:** a node on which specific types of components can be deployed on in the form of executable artifacts.
  - E.g. OS, workflow engine, database system, and JEE container.
- Execution environment instances are assigned to device instances.
- Execution environments can be nested (e.g., a database nested in an operating system).

Terminology

- Two terms that seem well-nigh interchangeable.

- **VM = virtual machine**
  - a software imitation of a physical machine/computer. It runs application software, using the physical resources of a host computer, but shields the application from having to know what computer is used and (likely) what operating system runs on that computer.

- **EE = execution environment**
  - a node that represents a particular execution platform, such as an operating system or a database management system. EEs can be nested; for example, a database EE can be nested in an operating system EE.