WebSphere User Group

Serviceability:

How Portal developers can turn administrators into friends AND have more time for development

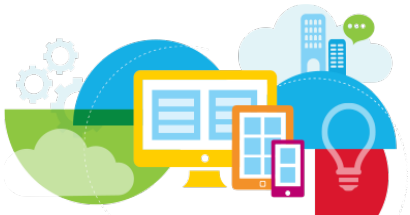
Richard Shooter, IBM Portal Infrastructure Architect
Graham Harper, IBM Portal Application Architect

Please Note

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion

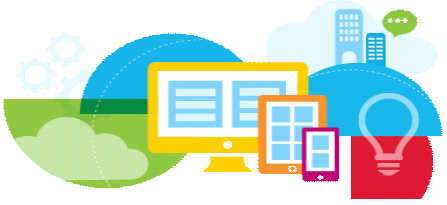
Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.



Agenda

- Introduction
- Serviceability
- Why serviceability matters to developers
- Addressing serviceability
  - Principles
  - Case study
  - Failures, operational issues and change
- Q & A

Introduction



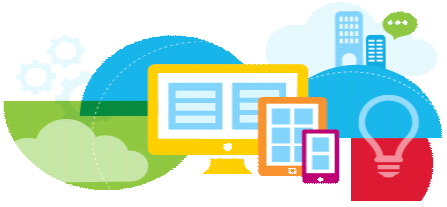
Who are we?

- Graham Harper – Portal Application Architect
- Richard Shooter – Portal Infrastructure Architect

We have a combined 20+ years of experience with assisting IBM customers with Portal in complex solutions and environments.

Some of that experience has been in assisting in resolving issues with those solutions or environments!

Serviceability



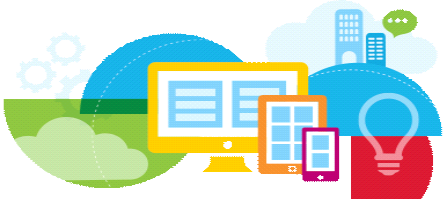
Serviceability

- The ability of technical staff to:
  - configure and monitor systems and services
  - identify exceptions or faults
  - perform root cause analysis
  - perform maintenance
  - restore systems into service

Non-functional requirements

- Qualities and constraints – often addressing areas such as:
  - Availability
  - Supportability
  - Configurability
  - Maintainability

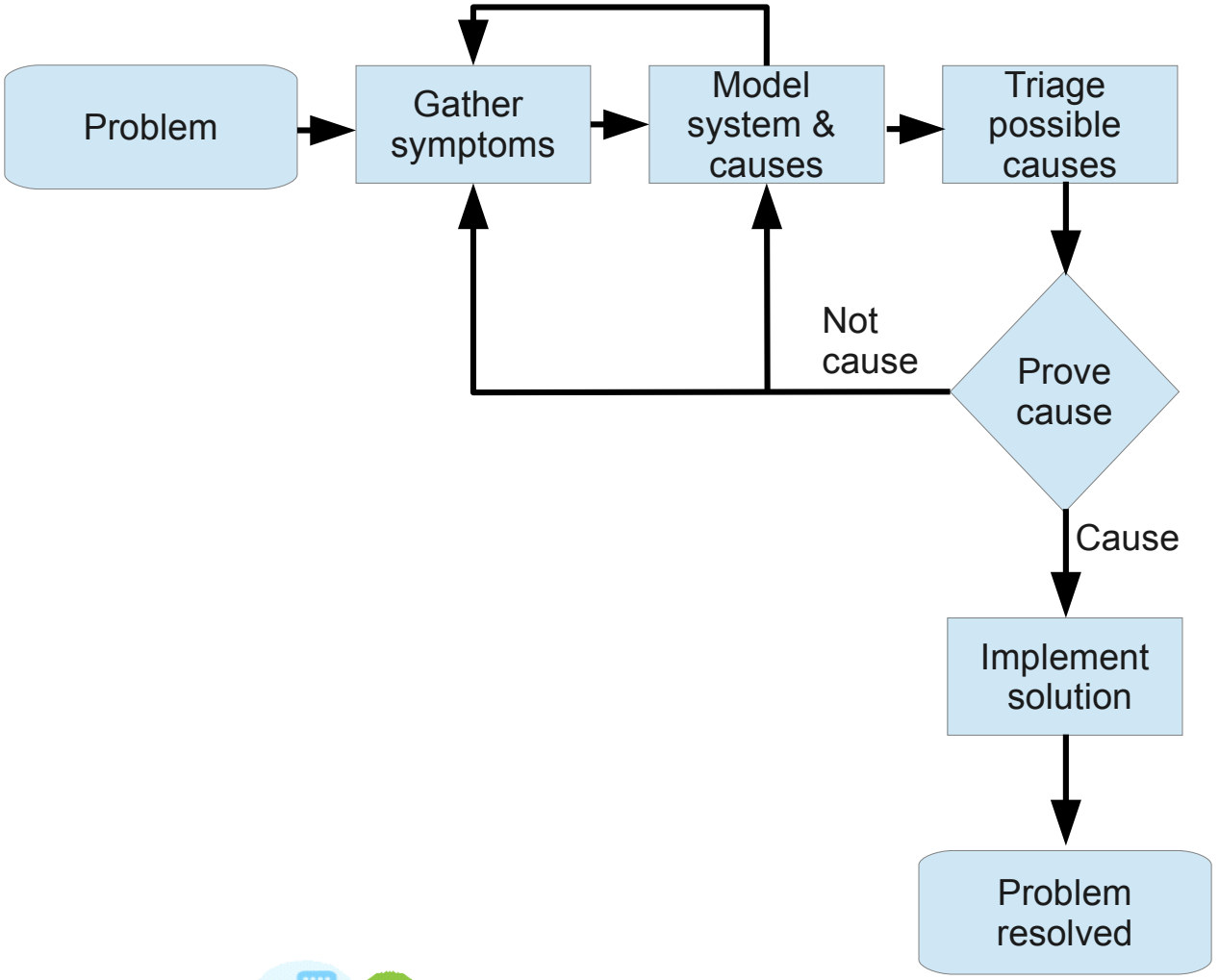
Why serviceability matters to developers



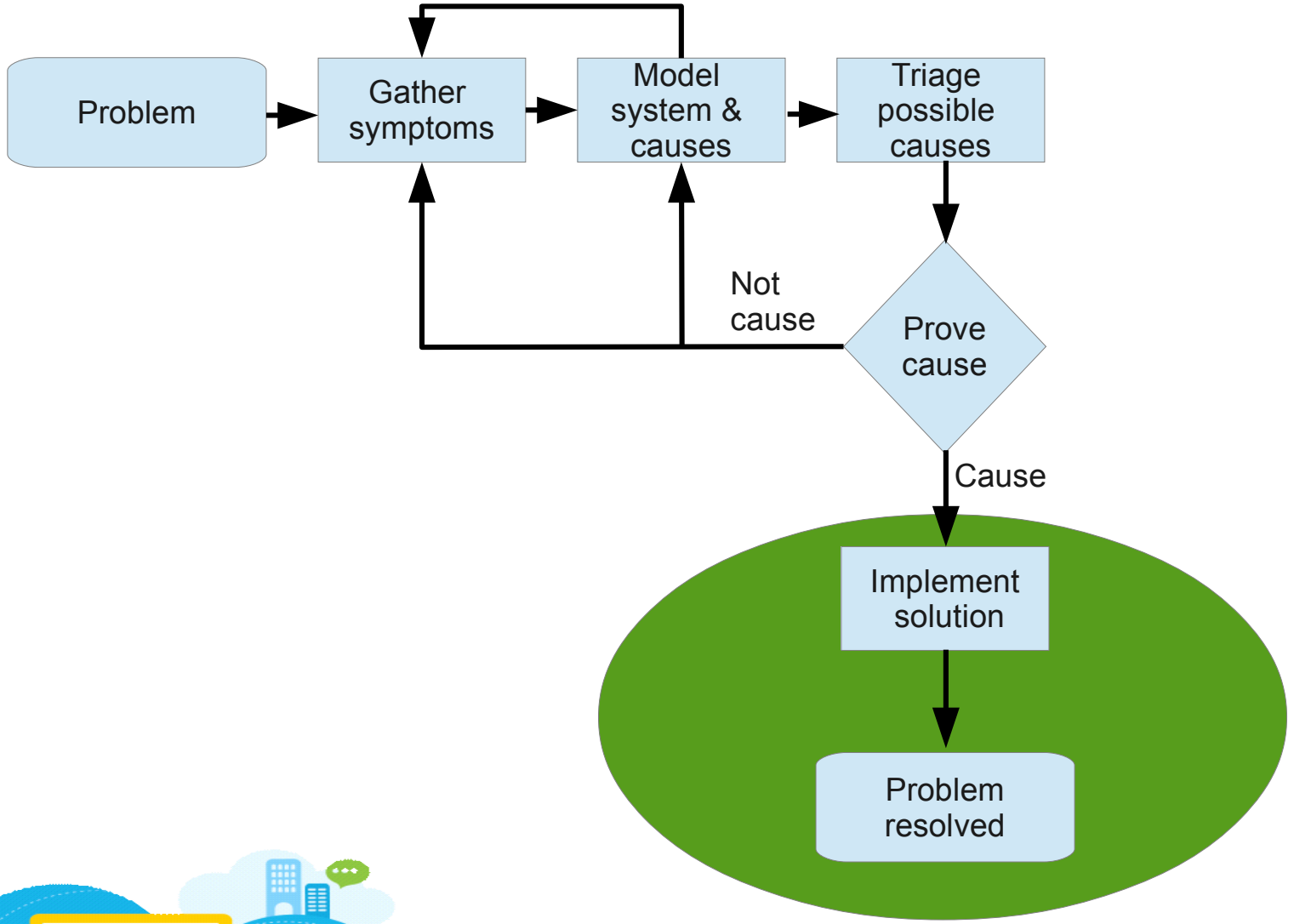
There is a “problem” with the production system

- This could be:
  - Availability related
  - A failure to meet other NFRs
  - Unexpected behaviour
  - Many other events
- Let's consider the process of investigating the “problem”

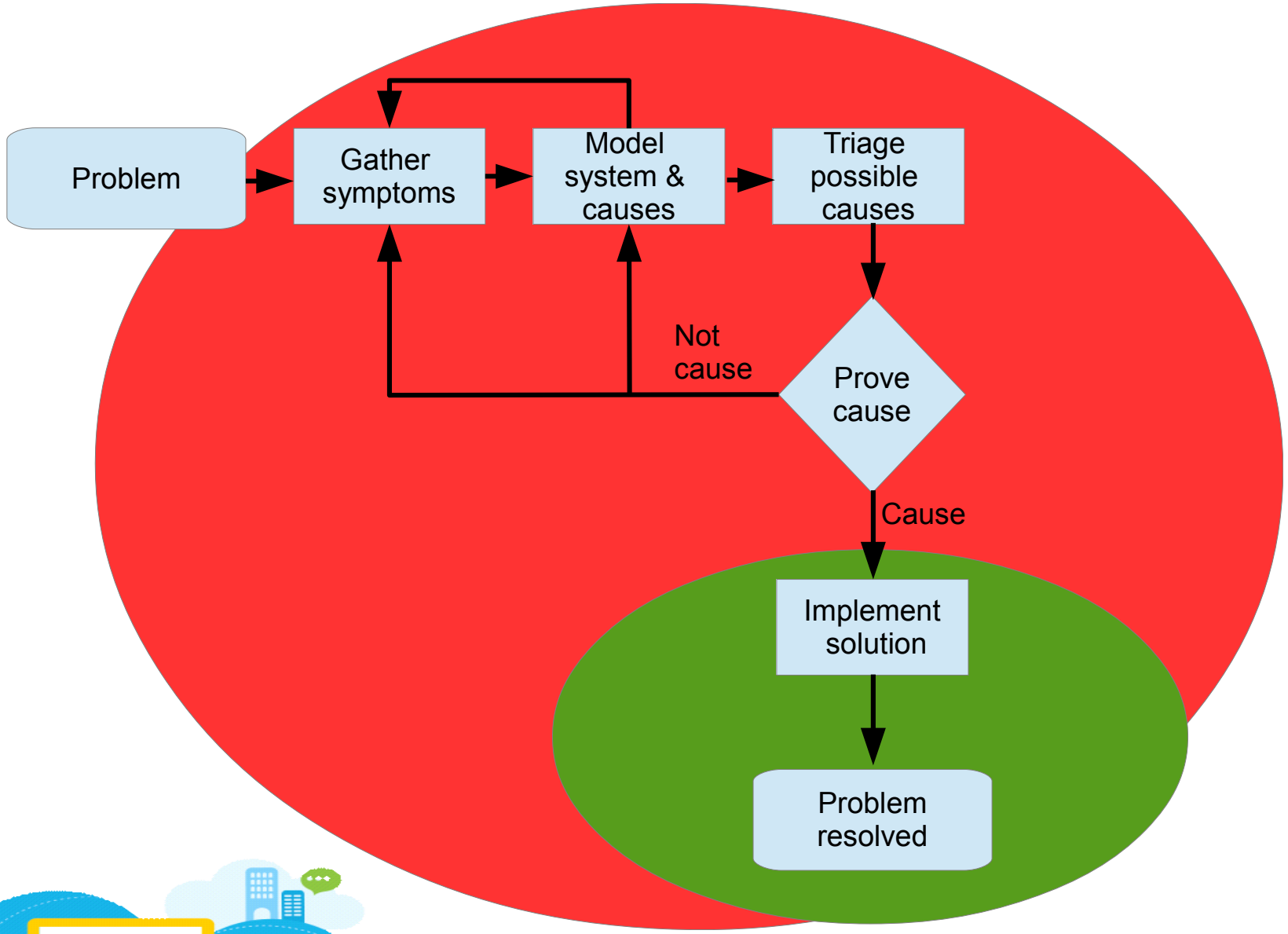
Problem investigation



When you want to be involved



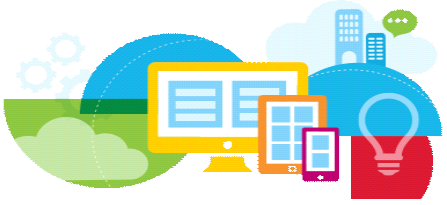
When you could be involved



How to minimise your involvement

- Avoid an incident becoming a problem – identify events which can be handled through BAU
  - Consider potential key events
  - Plan (design) for the events
- If an incident becomes a problem – should be quickly characterised and resolved without development involvement
  - Traceability
  - Documented solution design

Addressing serviceability: *Principles*



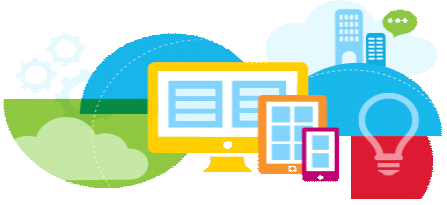
Availability – design considerations

- How much up-time is needed?
  - There is always a trade-off with cost
- Architect where necessary for:
  - Software component failure
  - Network and hardware failure
  - Maintenance and backups
  - Application deployments
  - Application server and operating system upgrades

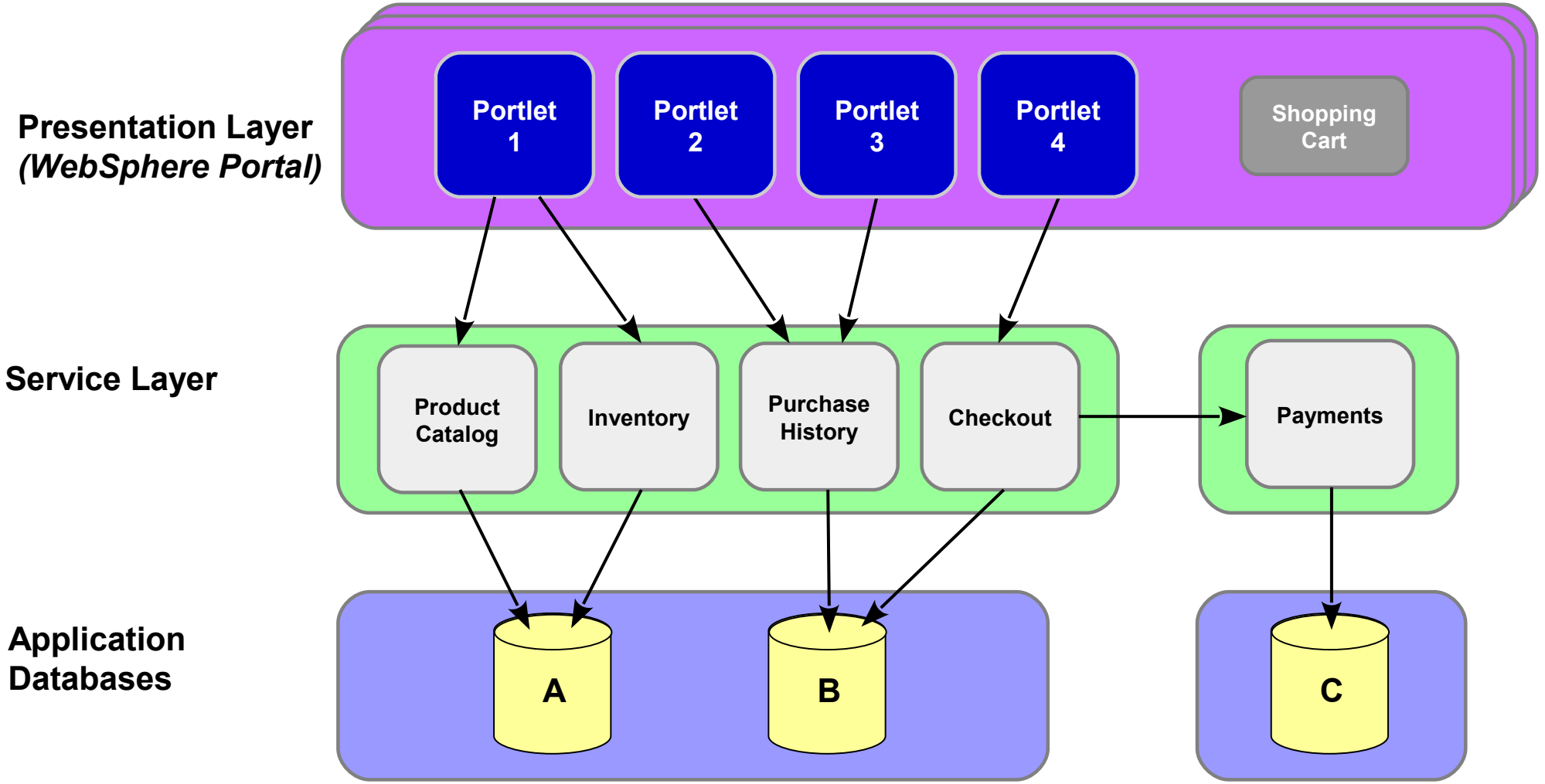
Reliability – design considerations

- Recover from errors where possible
- Limit impact of failovers and temporary outages
- Unaffected components should be usable
- Performance should be acceptable
  - and ***consistent***

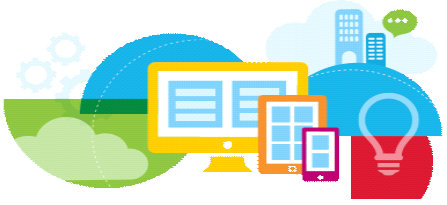
Addressing serviceability:
An (abstract) case study



An online shopping site



Addressing serviceability: *Failures*



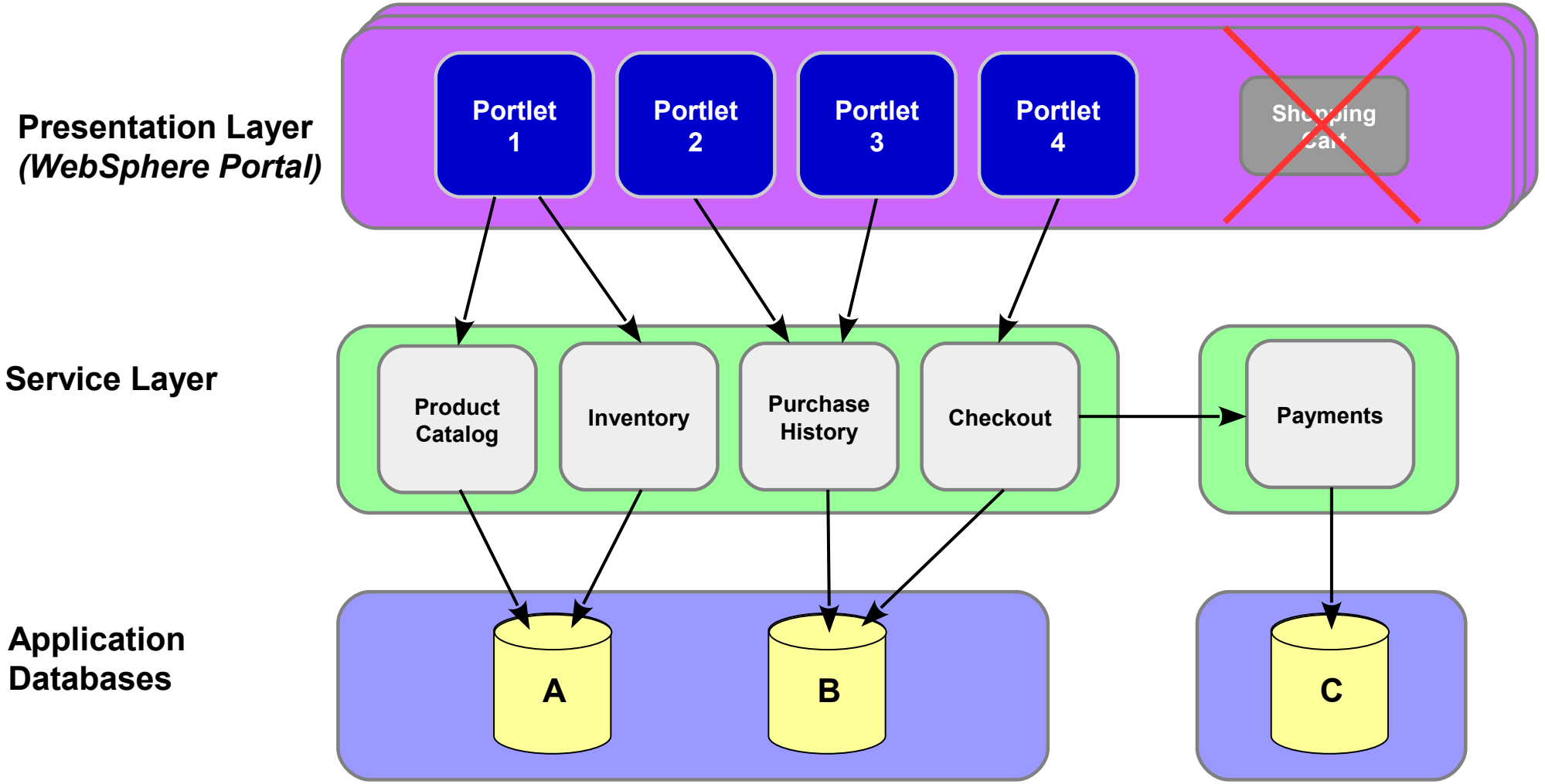
Failures

- ***Failures don't have to be developer problems***
  - If the application can self-recover
  - If the end users are unaware or minimally affected
  - If the support and operations teams can recover the failure without it becoming an issue

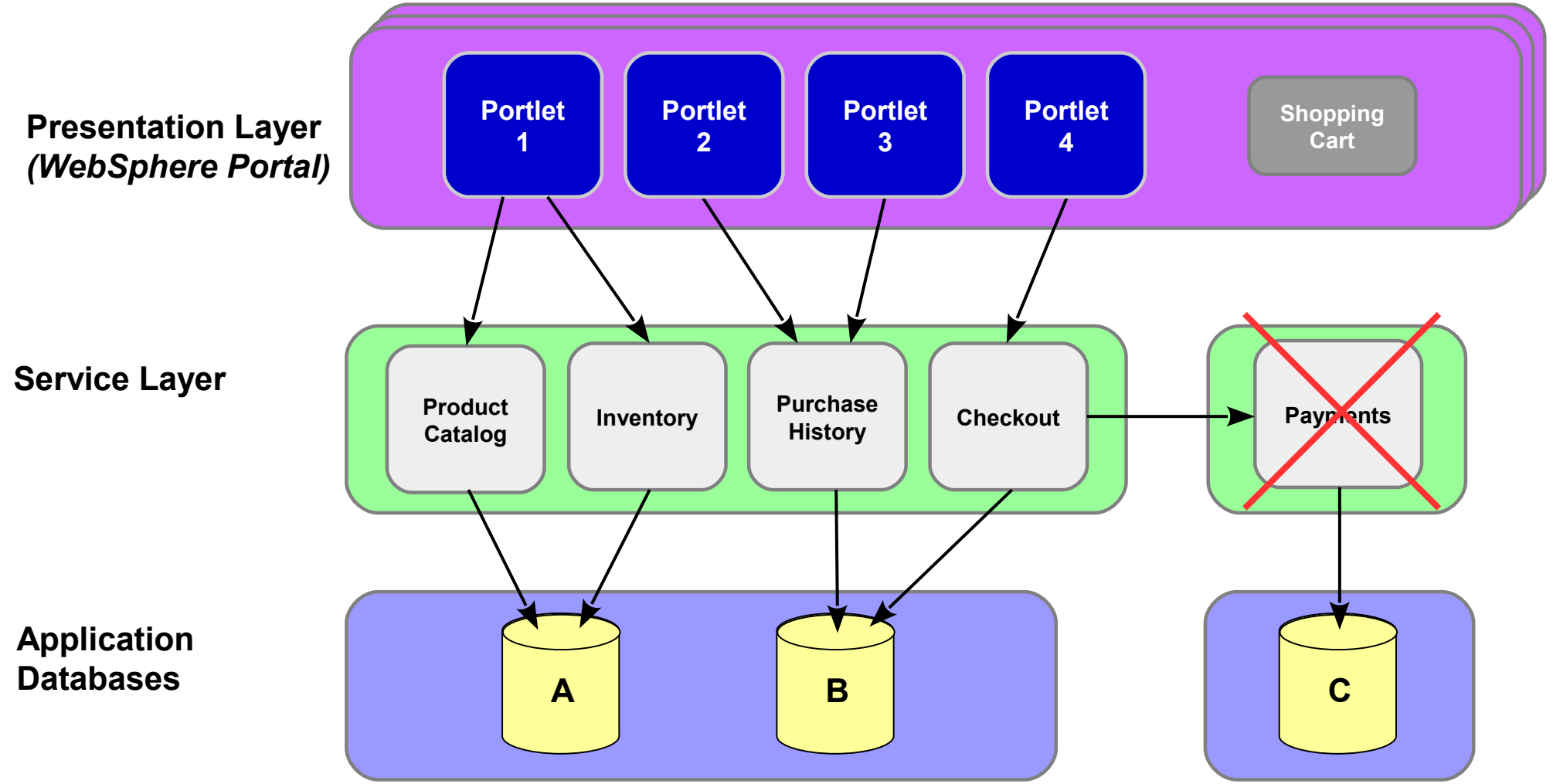
In our case study...

- What does the user see on failover due to Portal server hardware failure?
- Is their shopping cart lost?
- What about if a service is unavailable / returns an error? How do the portlets react?
- What do they see whilst we are deploying a new version of the application?

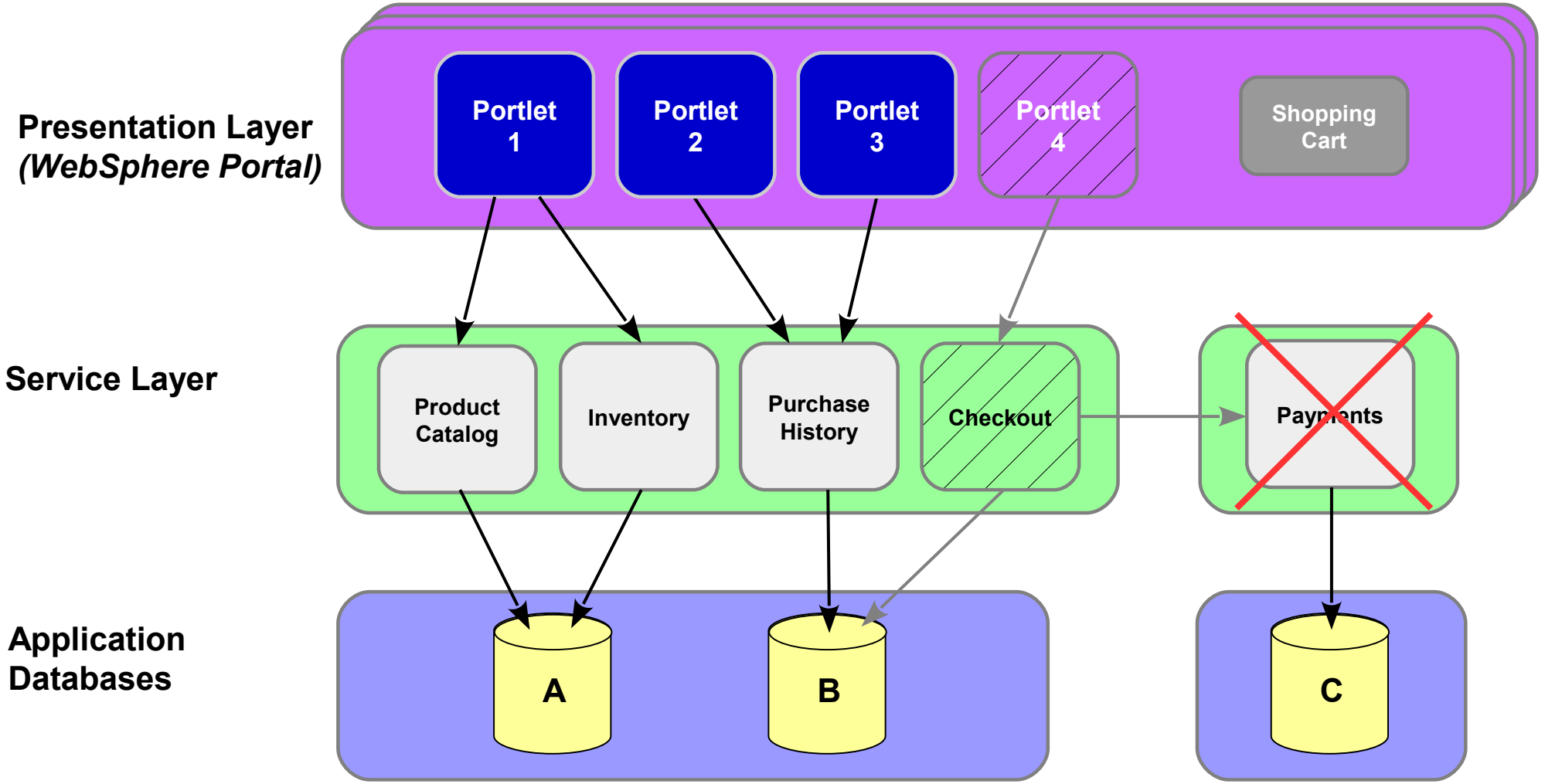
Shopping cart failover?



Service offline, but...



Partial functionality still available



Reliability

- Recover from errors where possible
  - Retry connection failures
  - Detect and refresh stale connections (e.g. JNDI lookups)
  - Keep session timeouts consistent or application must be aware of potential effects
- Fail over seamlessly – or at least informatively
  - Session replication (maybe)
  - Session recovery

Availability

- Unaffected components should be usable
  - If some functions are unavailable make them unreachable
  - Don't just redirect to an error page if some functions could still provide useful service
- Maintenance should not create failures
  - Maintenance windows
  - Updates without outages (see later)

Consistent performance

- Occasional very poor response times seem like failures to users
  - Anticipate legitimately long transactions
  - Warn users or keep apprised of progress
  - Consider asynchronous processing

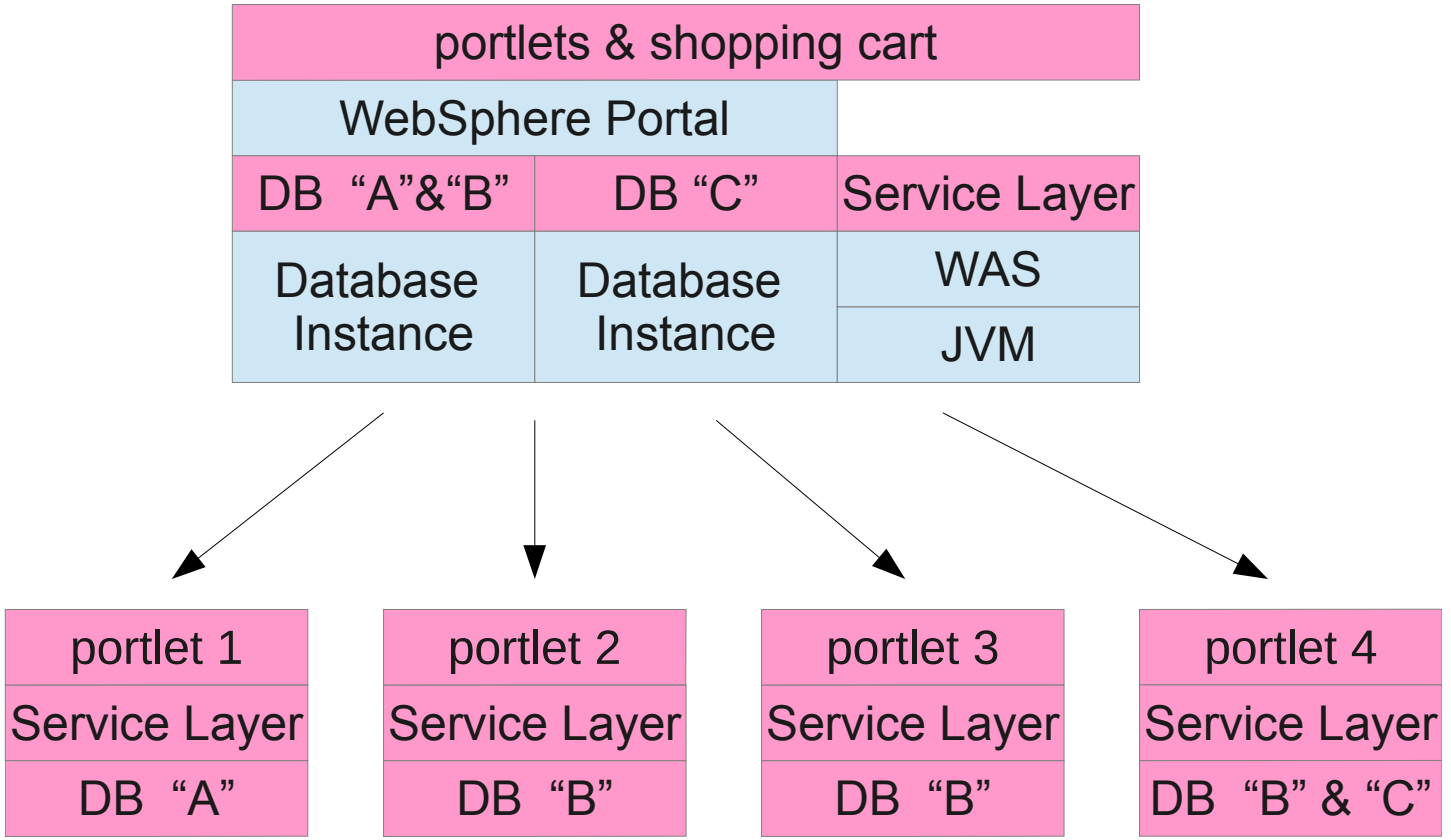
Generic Portal-platform service model

<table>
<tr><td colspan="2">HTTP Server</td><td></td></tr>
<tr><td colspan="2">portlet(s)</td><td></td></tr>
<tr><td colspan="3">WebSphere Portal</td></tr>
<tr><td rowspan="2">WebSphere Application Server</td><td>Config DB</td><td>Registry</td></tr>
<tr><td rowspan="2">Database Instance</td><td rowspan="2">Directory Server</td></tr>
<tr><td>JVM</td></tr>
<tr><td>Operating System</td><td>Operating System</td><td>Operating System</td></tr>
<tr><td>HW Image</td><td>HW Image</td><td>HW Image</td></tr>
<tr><td colspan="3">Hypervisor</td></tr>
<tr><td colspan="3">Hardware</td></tr>
</table>

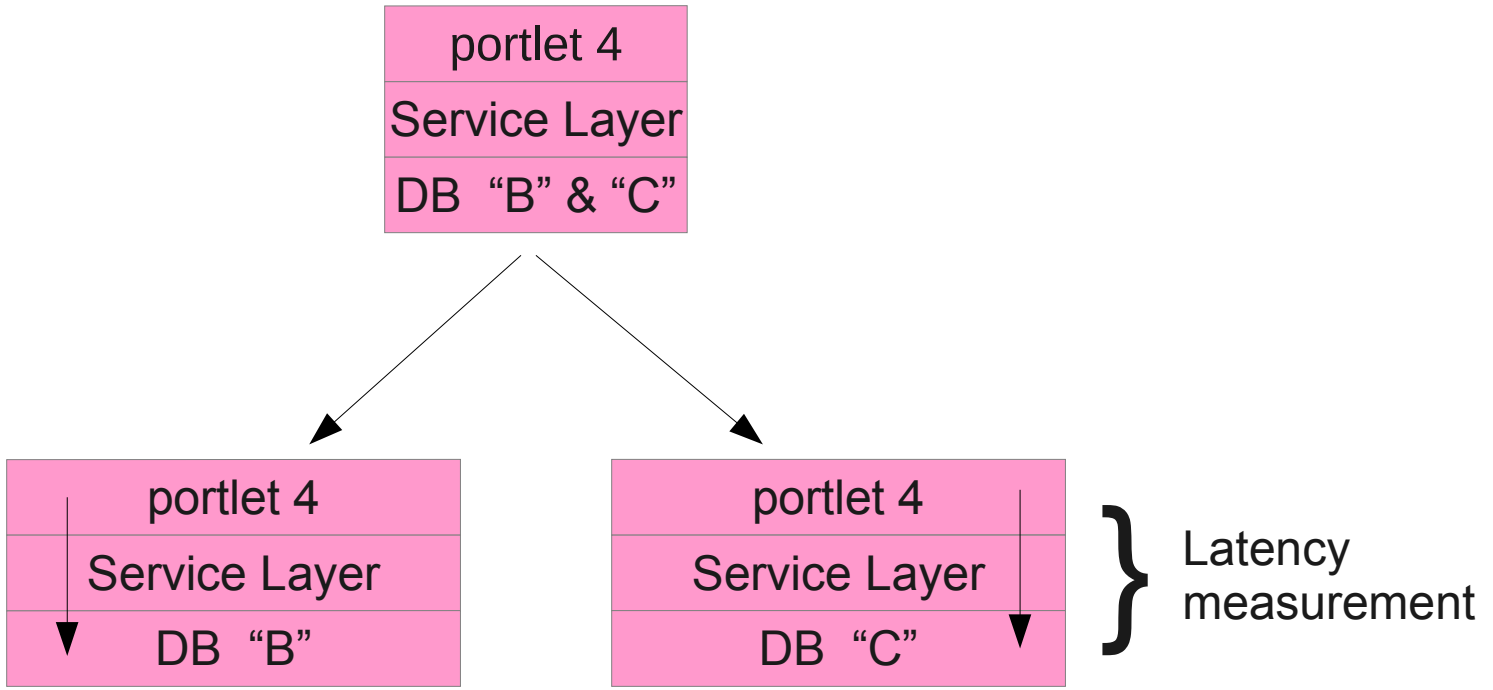
Solution service model

<table>
<tr><td colspan="6">HTTP Server</td></tr>
<tr><td colspan="3"></td><td colspan="3">portlets & shopping cart</td></tr>
<tr><td colspan="5">WebSphere Portal</td><td></td></tr>
<tr><td rowspan="2">WAS</td><td>Config DB</td><td>Registry</td><td>DB “A”&“B”</td><td>DB “C”</td><td>Service Layer</td></tr>
<tr><td rowspan="2">Database Instance</td><td rowspan="2">Directory Server</td><td rowspan="2">Database Instance</td><td rowspan="2">Database Instance</td><td>WAS</td></tr>
<tr><td>JVM</td><td>JVM</td></tr>
<tr><td>Operating System</td><td>Operating System</td><td>Operating System</td><td>Operating System</td><td>Operating System</td><td>Operating System</td></tr>
<tr><td>HW Image</td><td>HW Image</td><td>HW Image</td><td>HW Image</td><td>HW Image</td><td>HW Image</td></tr>
<tr><td colspan="6">Hypervisor</td></tr>
<tr><td colspan="6">Hardware</td></tr>
</table>

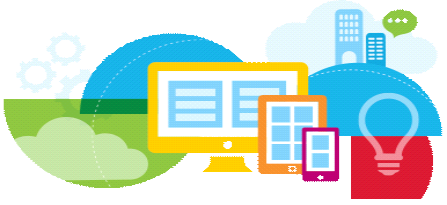
Considering the portlets



Performance monitoring



Addressing serviceability: *Operational issues*



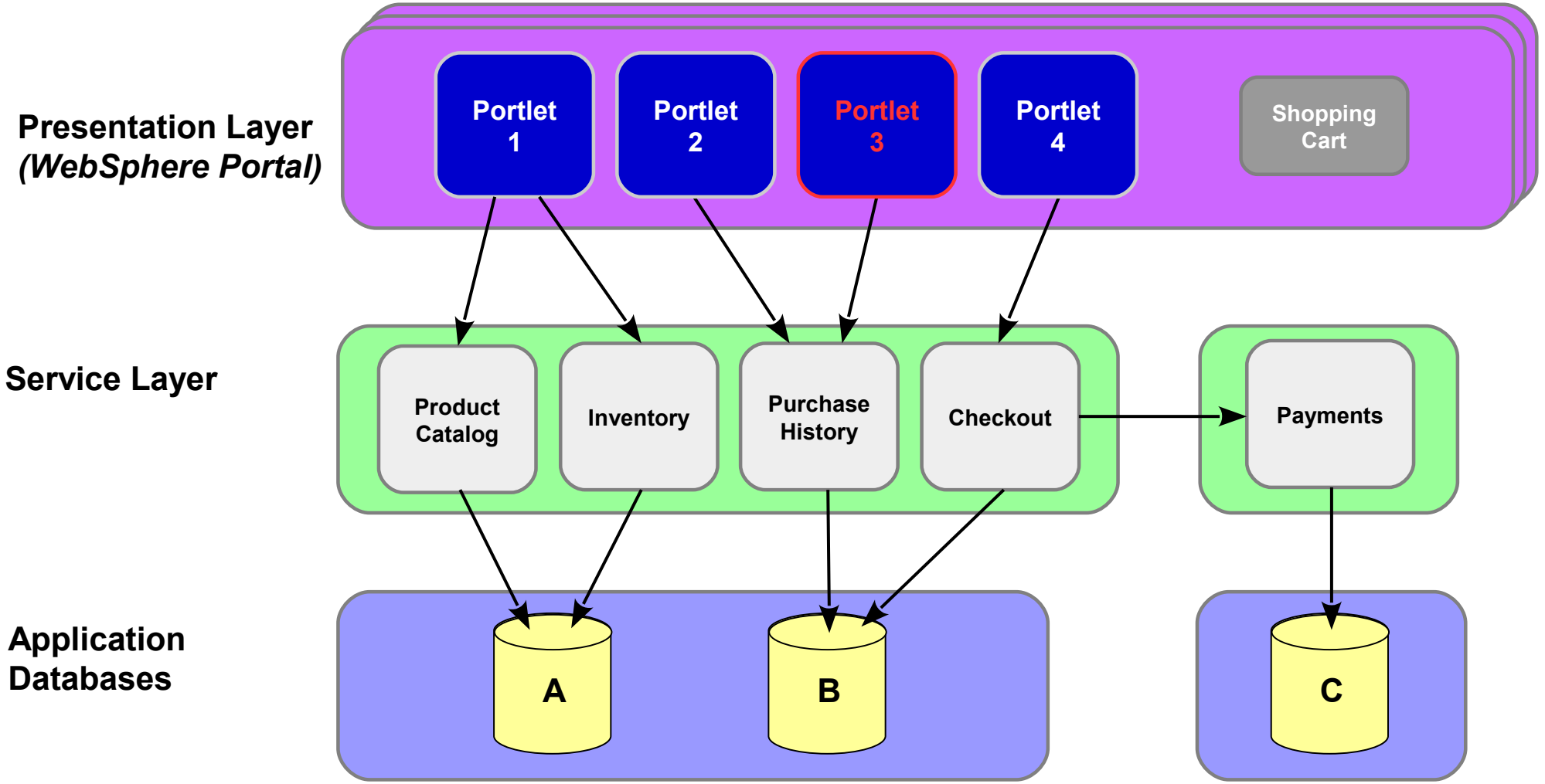
Sometimes failures will become issues

- ***Operational issues don't have to be developer problems***
  - If the application is *supportable*
  - If issues are *traceable*

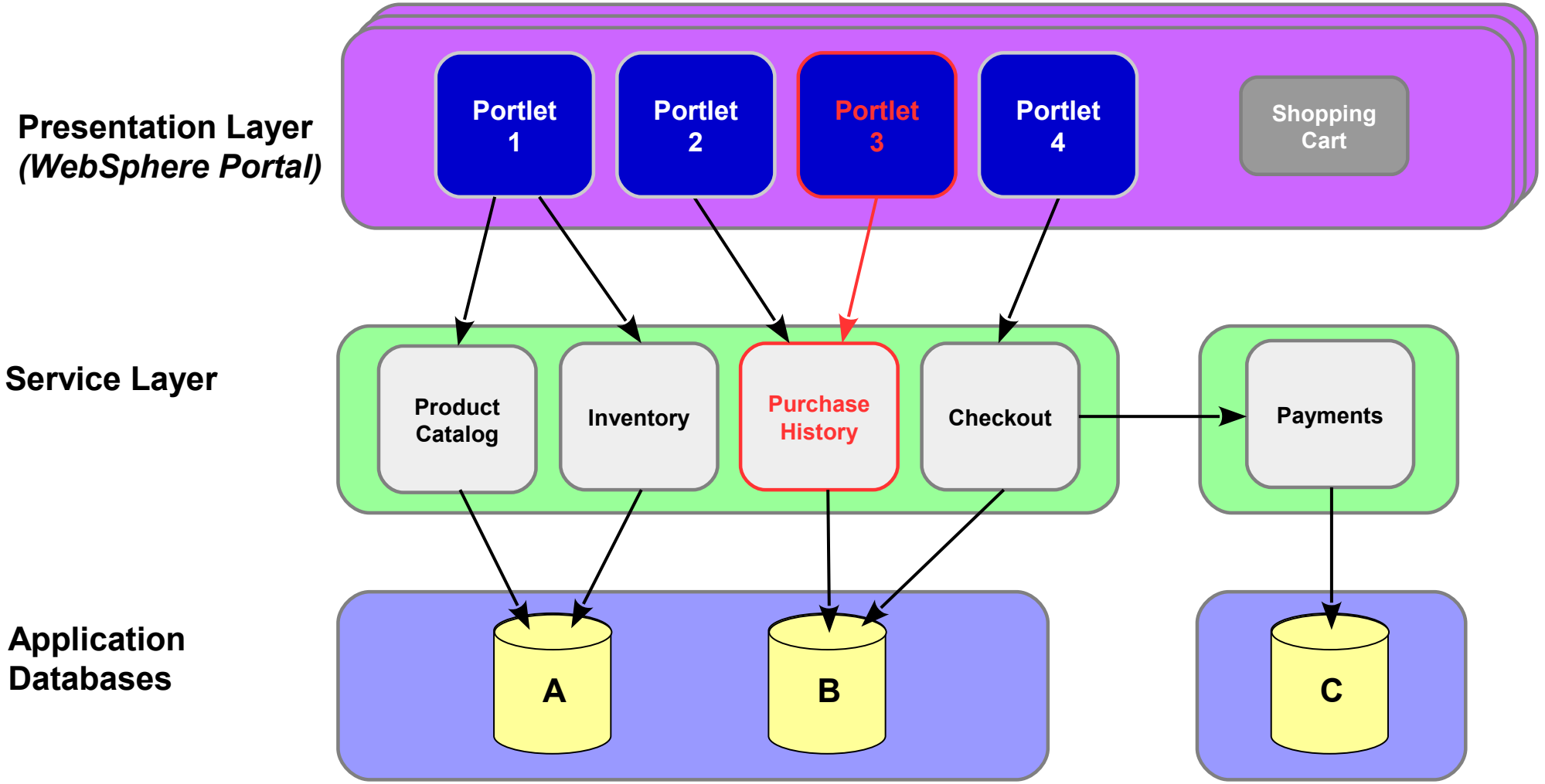
In our case study...

- What will the support desk say when a customer calls because they can't check out?
- Can we trace the failure reported by a portlet through to an underlying service or database issue?

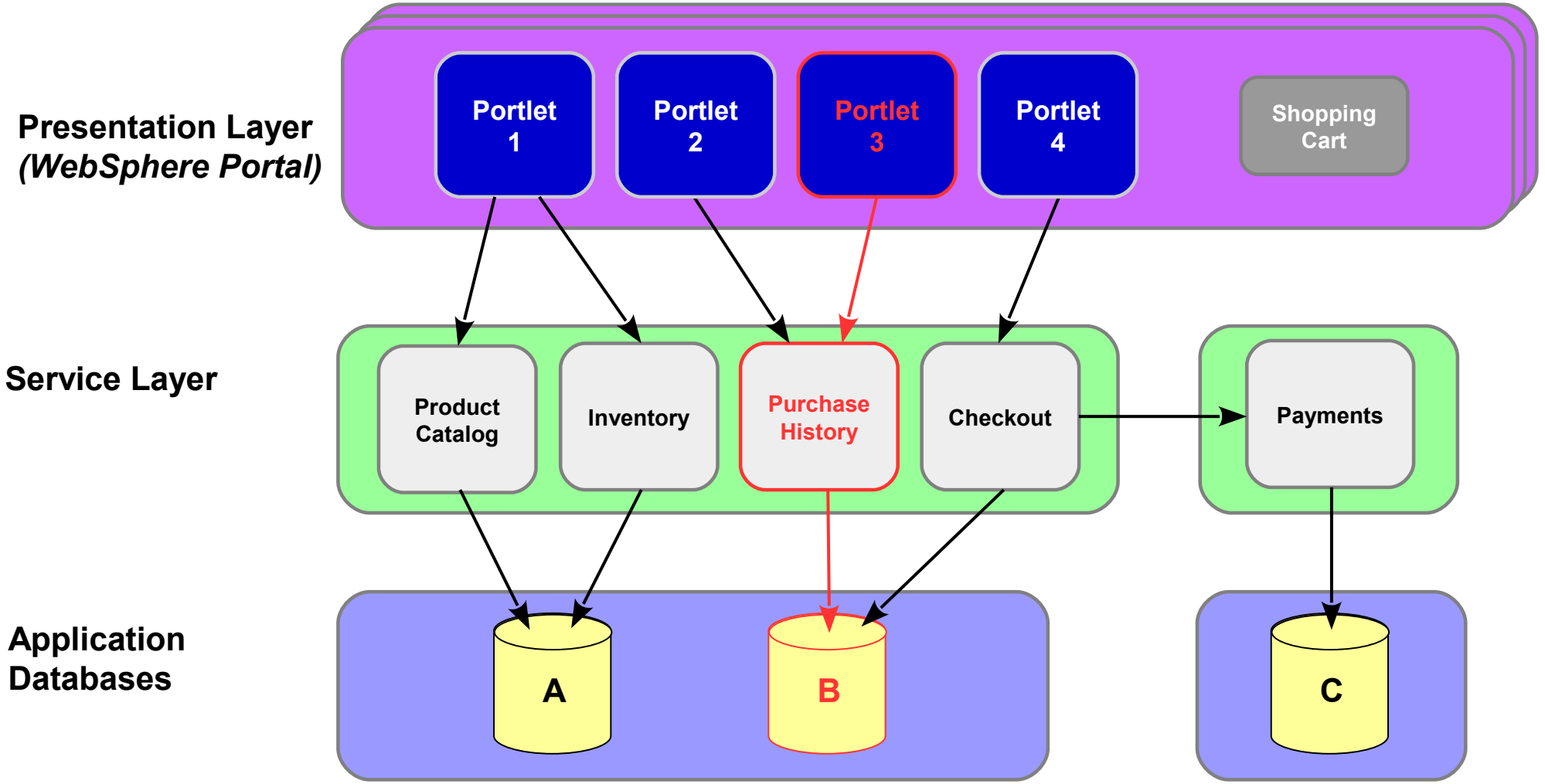
An online shopping site



An online shopping site



An online shopping site



Supportability

- Is customer support briefed on what the customer will see when a failure occurs?
- Can they distinguish:
  - Business issues (customer's card declined)
  - Intermittent issues (network problem, node down)
  - Systemic issues (layer unavailable, data corruption)
- Are we reliable enough that failures are rare?
- Do we have appropriate escalation levels and paths?

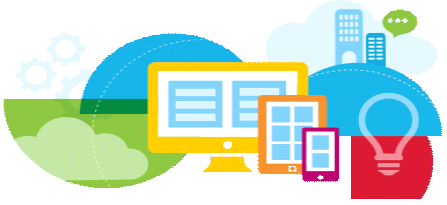
Traceability

- Can we follow a problem through?
  - e.g. unique user session and /or transaction IDs across layers
- Stack traces are rarely informative error log messages!
- Can we easily configure additional logging?
- Can we enable performance measuring?
  - Problem might be poor performance for some users and not complete failure

Operational control

- Do the operations team have the ability to control necessary aspects of the application?
  - Monitor, clear and re-size caches
  - Change pool sizes
  - Modify numbers of threads
- See “configurability” later for mechanisms

Addressing serviceability: *Change*



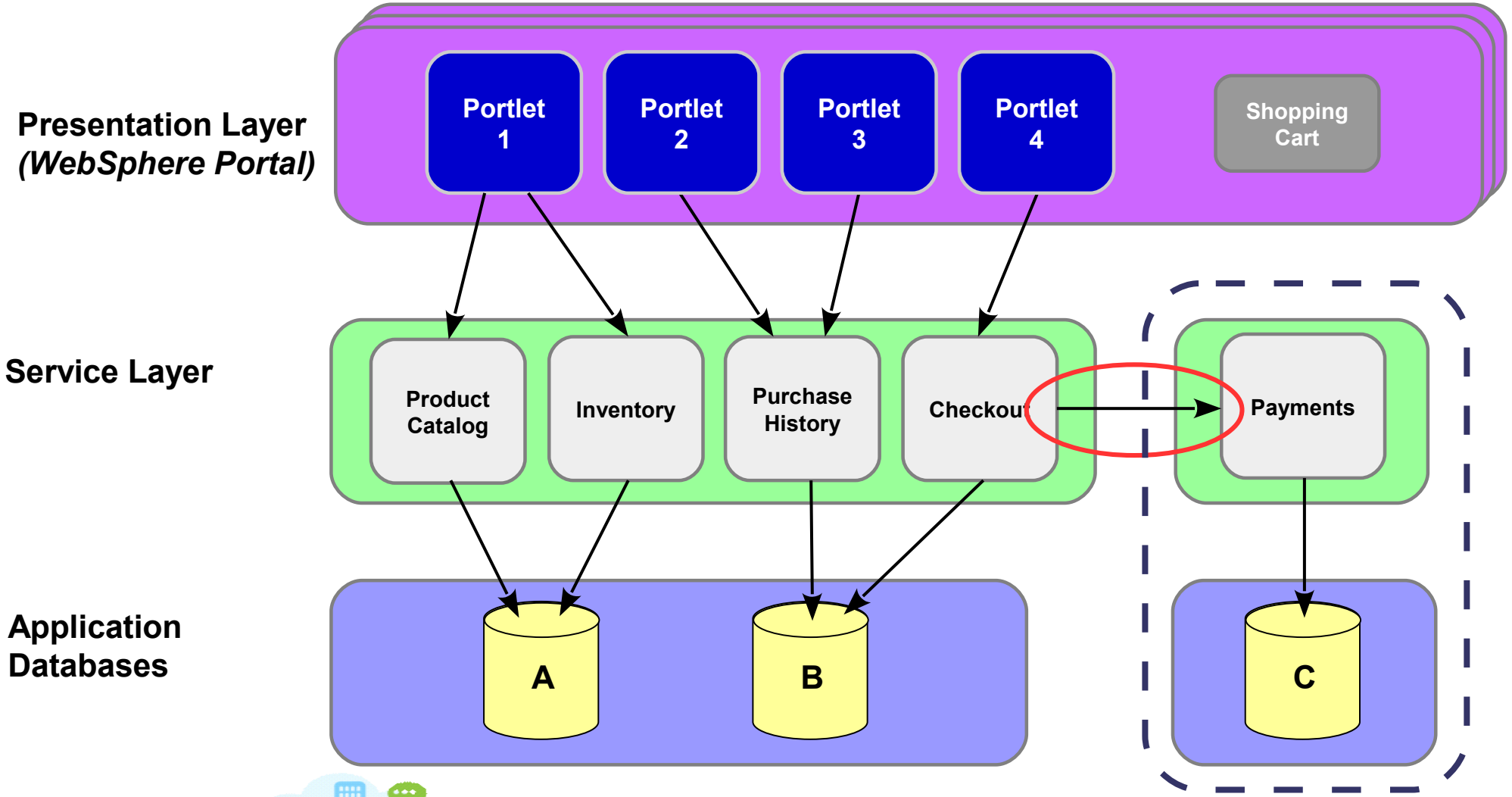
Change

- ***Change doesn't have to be a developer problem***
  - If the application is easily *configurable*
  - If it is *maintainable*
    - Legitimate developer *work* should not become a developer *problem*

In our case study...

- If a configuration setting (e.g. an external URL) changes, what needs to happen?
  - Nothing, it is picked up immediately
  - Nothing, it will be picked up in minutes / hours
  - Server restart
  - Application redeployment

An external system moves



Configurability

- Where are your configuration settings?
  - Constants in a Java class
    - Requires application redeployment
  - Resource Environment Provider
    - Requires server restart to pick up new value
  - Configuration file / database
    - Application **can** re-read periodically
  - JMX objects
    - Change on the fly for immediate effect

In our case study...

- You need to make a small change to a portlet and / or service: how do you re-test?
  - Re-run automated regression tests
  - Get out the manual test scripts and go through them
  - Prod things a bit and hope

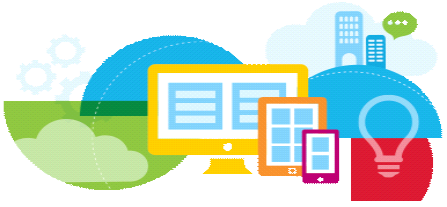
Maintainability

- Regression tests
- Sensible application partitioning to reduce impact of changes
- Layering with well-defined interfaces
- Deployment scripting
- Incremental deployment
  - To preserve customisations
  - Release Builder

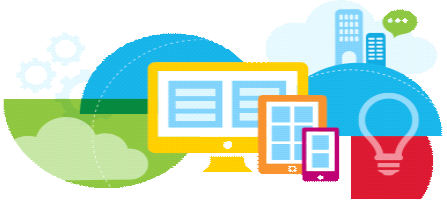
Continuous improvement

- Refactoring for better serviceability
- Incorporate feedback and experience on the -ilities into new versions

Questions?



Thank you!



For Additional Information

- **IBM Digital Experience Solutions**
  http://www-01.ibm.com/software/collaboration/digitalexperience
- **WebSphere Portal and IBM Web Content Manager Information Center Wiki**
  http://www-10.lotus.com/ldd/portalwiki.nsf/
- **IBM Digital Experience Demonstrations:**
  http://www.youtube.com/user/IBMXWebX
- **IBM Collaboration Services Business Solutions Catalog**
  https://greenhouse.lotus.com/catalog/