

**Maximize Availability.  
Minimize risk.  
Even during high throughput.**

The most critical feature of a system that keeps the bank up and running is the resilience of the core banking system. Zero downtime means that you can service customers of the bank, on failure of any component in the system.

Finacle is continuously investing time and effort to increase the availability of the solution. It has been made to operate in live environments with a number of other supporting systems, till such point that 24/7; resilience, availability and scalability have become standard features of Finacle.

#### Built for availability

Finacle universal banking solution provides near-continuous availability for hardware or software component failures even during high throughput. Finacle has been thoroughly tested and demonstrated such that there is no loss of transactions during hardware or software component failures. **In some component failure there is zero failed transaction and in other cases where the transactions are in process at the time of failure, the transactions are replayed to completion.**

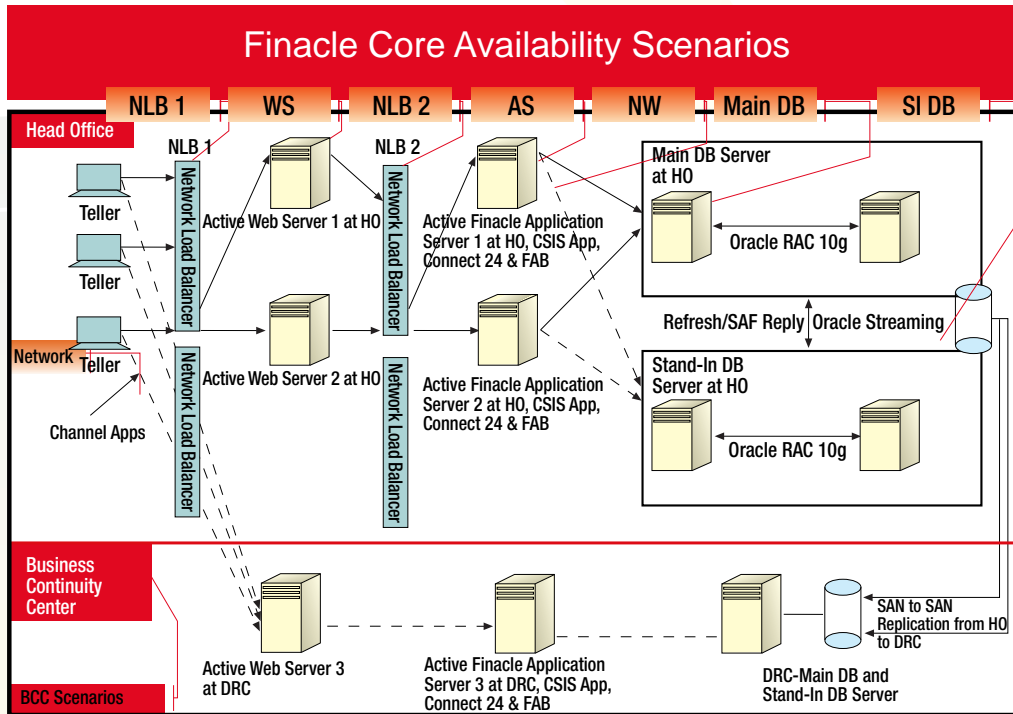
#### Proven to minimize downtime in real life situations

During a demonstration in partnership with HP, Radware and Oracle, (conducted at HP Performance Benchmark Center at Boeblingen, Germany), Finacle was subjected to 31 different failure scenarios. Finacle proved to be highly resilient with minimal interruption due to hardware and software component failures. **The failures were either none or less than 0.01% at high throughput.**

#### Highlights

- There was no loss of transaction at high throughput on failure of hardware or software component.
- Downtime, planned or unplanned, for maintenance or created by a server crash or disaster resulted in zero or near zero outages.
- Bringing up Finacle services after component failures recorded less than a minute for servers across the architecture tiers.
- During the demonstration Finacle was deployed in a fault tolerant architecture with no single point of failure. The deployment architecture had sufficient redundancy built-in to ensure application availability in the event of any failure.
- Availability during end of day operations was 100% with normal branch and delivery channel transactions available round the clock.
- There was no impact of online backup on Finacle.
- Refresh to stand-in server from main database was completed in a minute after one million transaction upload.

The following diagram depicts the three tier architecture of Finacle deployment for high availability.



The following types of failure were simulated for the demonstration.

- Network failures (NIC, LAN, load balancer failures) at web layer and application layer:** Different failures were simulated such as disconnecting the network between the application and database server, powering off the primary load balancer and powering off all the LAN network switches. All these had nil or minimum outages.
- Server failures at the web, application and database layers:** Individual servers were crashed during a high throughput run, to check impact. There were very few transaction failures that were executing at the point of server failure. Finacle users have a choice to re-enter or re-submit the transactions.
- Database failures:** One of the real application cluster instance was failed during the high throughput run. All the database connections failed over to the remaining database instances successfully without any failure.
- Finacle application component failures at web and application servers:** Finacle application components like Finacle integrator and Connect 24 were failed on one web or application DB server during the high throughput run. The transactions were successfully transferred to the other server.
- Disaster situations at the main data center:** Disaster situations were simulated by shutting down all the database servers at the head office. The database was recovered successfully at the disaster recovery center. Users were able to continue operation from the disaster recovery setup.

