

# TEST STRATEGY

# Agenda

## 01 SCOPE

---

## 02 TESTING ACTIVITIES

---

## 03 TEST PHASES

---

## 04 TEST LIFECYCLE

---

# Agenda

## 05 TESTING FRAMEWORK

---

## 06 TEST ENVIRONMENT STRATEGY

---

## 07 AUTOMATION STRATEGY

---

## 08 DEVOPS STRATEGY

---

# Agenda

## 09 SKILLS REQUIRED

---

## 10 TEST ESTIMATION APPROACH

---

## 11 ENTRY AND EXIT CRITERIA

---

## 12 TESTING MEASUREMENT AND MATRICES

---

# Agenda

## 13 DEFECT MANAGEMENT

---

# Scope

## **In Scope:**

- Test Planning
- Test Case Design
- Functional and Regression Testing
- Automation Testing
- End to End SIT
- UAT Support
- Smoke Testing in PROD

## **Out of Scope:**

- UAT Testing
- PROD Testing
- Performance Testing
- Security Testing

# Testing Activities

## Testing Activities in Different Phases of Delivery:

- **Plan & Initiate** (Inception)
- **Design/Build/Validate** (Implementation & Stabilization)
- **Acceptance Test & Deploy** (Stabilization & Deployment)

Plan & Initiate	Design/Build/Validate	Acceptance Test & Deploy
<ul style="list-style-type: none"><li>• Test Approach/Strategy</li><li>• Test Automation Approach/Strategy</li><li>• In Scope/Out of Scope</li><li>• Entry &amp; Exit criteria</li><li>• Test Environment Strategy</li><li>• Testing Metrics</li></ul>	<ul style="list-style-type: none"><li>• Sprint Planning</li><li>• Functional Testing</li><li>• Stabilization/Cross Workstream Testing</li><li>• Entry &amp; Exit criteria</li><li>• Smoke Testing</li><li>• API/Integration Testing</li><li>• SIT/E2E Testing</li><li>• Regression Testing</li></ul>	<ul style="list-style-type: none"><li>• UAT</li><li>• Post Deployment validation</li></ul>

# Test Phases (1/2)

Testing is a structured way of validating requirements and specifications are properly implemented in a solution, proving whether or not the solution meets customer's functional, technical, operational and maintenance expectations.

In order to manage the complexity of testing the solution, testing will be divided into a number of standard phases, each with specific objectives and scope.

## **Test Phases structure:**

- Sprint Test (ST)
- System Integration Test (SIT)
- User Acceptance Test (UAT)
- Production Test (PROD)

SIT can be started after 3 to 4 Sprints are over



# Test Phases (2/2)

The **objectives and scope** for each test phase are reflected in the following table:

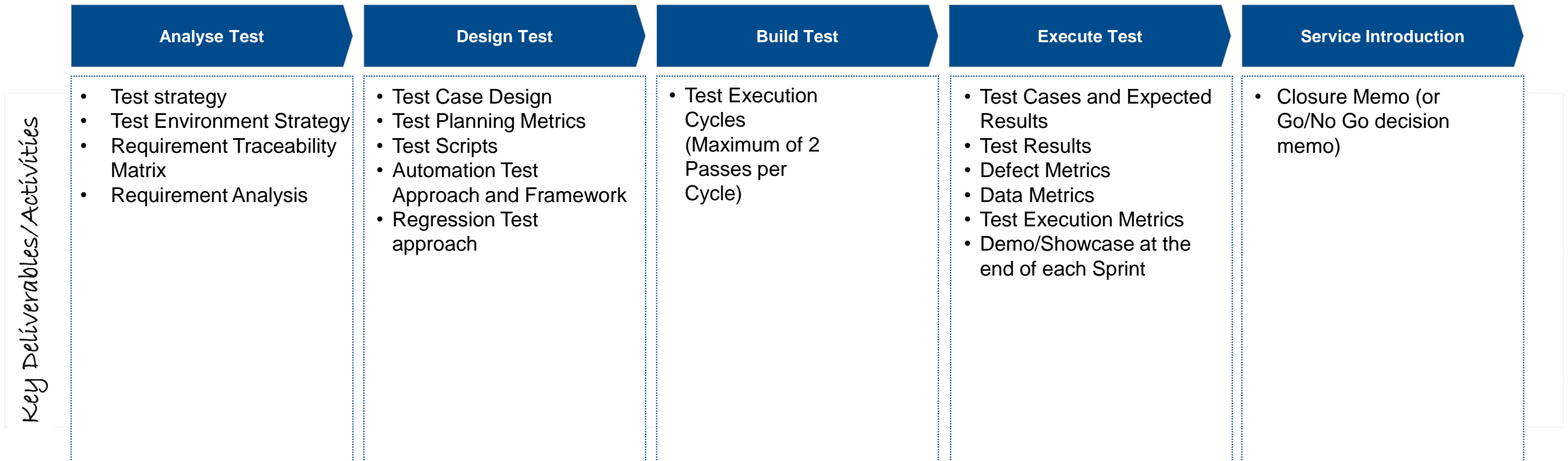
	Sprint Test	SIT	UAT	PROD
Objective	<ul style="list-style-type: none"> <li>Functional and Integration Testing of the requirements</li> </ul>	<ul style="list-style-type: none"> <li>System Integration Testing phase, which is more focused on functional end-to-end processes. This phase ensures that the system meets the business requirements</li> </ul>	<ul style="list-style-type: none"> <li>Identify and correct functional errors on process level and ensure that the system meets the functional requirements</li> </ul>	<ul style="list-style-type: none"> <li>Running a series of fast and basic tests on the core functionality to ensure deployments are successful</li> </ul>
Scope	<ul style="list-style-type: none"> <li>Validate that all Acceptance criteria are met and all Product/Sprint Backlogs are cleared</li> <li>API Testing</li> </ul>	<ul style="list-style-type: none"> <li>Validate that collection of systems work together and cross-application business requirements are met</li> <li>Validate application interfaces between adjacent systems are functioning correctly</li> </ul>	<ul style="list-style-type: none"> <li>End-to-end test performed by the business to validate that the application meets user requirements and supports business processes</li> </ul>	<ul style="list-style-type: none"> <li>Performing Smoke testing (only view page) to ensure the connections are intact in Production environment to make sure the product is ready for further testing</li> </ul>
Testing	Manual & Automation	Manual & Automation	Manual	Manual/Automation

## Automation Testing Framework

GT-UI/GT-API Framework can be leveraged for UI, API and Regression Testing during System Integration Testing

# Test Lifecycle

Standard test phases are **divided into stages**, where a test stage is a collection of related process activities grouped together along the testing life cycle.



# Testing Frameworks (1/2)

Testing Area	Testing Tool/Framework	Testing Phase
Smoke	Manual OR GT:UI & GT:API	Sprint
	GT:UI & GT:API	SIT
	Manual	UAT
Sanity	Manual	Sprint
Functional	Manual	Sprint
	GT:UI	SIT
	Manual	UAT
E2E	GT:UI	SIT
	Manual	UAT

# Testing Frameworks (2/2)

Testing Area	Testing Tool/Framework	Testing Phase
API/Integration	GT:API	Sprint
	GT:API	SIT
	GT:API	UAT
Regression	GT:UI & GT:API	Sprint
	GT:UI & GT:API	SIT
	Manual	UAT
Performance	NA	NA
Cloud Security	NA	NA
Application Security	NA	NA

# Test Environment Strategy

- Guidewire should provide dedicated All Testing Environment for Dev and QA
- Testing Environment should only be updated with code base after a sprint/release
- There should be different QA environments each for Sprint, SIT, UAT, PROD, CONV, Automation and Performance Testing
- Should have QA environments both for Non-Time-Based Testing (NTBT) and Time-Based Testing (TBT)
- Test manager to ensure that test environments are before the start of testing  
Any delay in this may impact the effort required to complete the testing & revisit timelines
- The Testing Team will perform a Smoke Test to ensure that the test environment is ready for test execution
- Configuration Management/Environment Support team would inform the test team well ahead of time for any scheduled outages  
Unplanned outages may result in downtime which impacts effort & cost
- DevOps team will be responsible for network connectivity between systems, and they will also simulate the production batch jobs (if any) into new test environments to support Integration and System testing

- **Automation Strategy is required UI and API Testing:**  
UI automation to be done using GT:UI  
API automation to be done using GT:API
- **Design consists of primarily two major activities (high-level design and low-level design):**  
High-level design confirms the Test automation architecture  
Low-level design activity helps to identify the various business components and business flows, which form part of the reusable libraries
- **Test Script Deployment:**  
Once the Test Script is assembled, Test Data is populated, Test scripts are validated then it is deployed in the Test Environment using CI/CD tools
- **Maintain & Closure:**  
Due to Requirement changes or Defects, the automation scripts would need updates on regular basis due to changes in user interface, business logic and workflow
- **Environment Strategy:**  
A separate environment is required to have continuous automation scripting and execution
- **Automation Execution Strategy:**  
Execution of the automated scripts will be done based on the scope of testing  
Execution should start and stop based on the entry and exit criteria
- **Risks & Mitigations Strategy:**  
Risks, mitigations and contingencies should be planned and documented. Assumptions for the risks should be taken into consideration.

# DevOps Strategy (1/2)

## Test Environment Setup:

- DevOps team needs to setup the test environments as per the requirements for each.
- They need to upload the master data/initial test data into the test environments after setting up the test beds.

## Test Environment Setup:

- Once builds are ready from the development team, builds need to be deployed in the testing environments without any delay.
- Code deployments should be scheduled (once in a day/twice in a day etc) so that testing team can plan their activities accordingly without getting much impacted for the downtime.
- Proper communication needs to happen on the notification of the build details specifying the items got deployed in that specific build, or defect ids deployed, build numbers, applications impacted etc.
- Downtime due to code deployment should be tracked by the team.
- Instant communication to the test team on availability of test environment after code deployment should happen.
- If manual Smoke tests are required than that needs to be communicated to the test team as soon as the code deployment is completed. Test environment needs to be on hold for any usage during the progress of Smoke Testing.

# DevOps Strategy (2/2)

## CI/CD Pipeline Maintenance for Automation Tests:

- Automated Smoke tests need to be run as soon as deployment gets completed. Automated pipeline should be maintained, and automation notification can be sent out once automated smoke test suite is completed.
- Any failed smoke tests need to be communicated at the earliest to the test team. Can be automated as well.
- Automated regression tests to be run as and when required from the CI/CD pipeline. Failures to be reported to the test team.
- Multiple servers can be used for running the automated regression suite to optimize the running time.

## Ad Hoc Requests:

- In case a database drop is requested, then it should be performed for the applications required. Approval of DB drop should be given by the owner of the test environments.
- In case of any urgent defect fix deployment is required, then it should be performed, and proper communication should be sent out.
- Any specific test data is required, then it should be loaded in the test environments.



# Skills Required

Role	Required Skills
<b>Test Manager</b>	Agile Delivery & Closure
	GWPC
	P&C Insurance Domain
	Test Planning & Strategizing
	Testing Environments
	Test Estimation & Resourcing
	Risk Assessment & Mitigation

Role	Required Skills
<b>Functional Test Architect</b>	Agile Test Delivery
	GWPC
	P&C Insurance Domain
	Test Management
	Test Design & Execution

Role	Required Skills
<b>Automation Test Architect</b>	Automation Strategies & Tools
	GWPC
	API & UI Test Automation
	Cucumber & Karate
	Typescript & Node JS & Gradle

# Test Estimation Approach (1/2)

- **Factors to be considered in Test Estimation Process:**

- Complexity of the Application

- Availability of requirements

- Availability of Testing Artefacts

- Proper access to external applications

- Prior Experience

- Skillset & domain knowledge

- Performance of the application

- Tools used for testing

# Test Estimation Approach (2/2)

- **Pre-requisite for Test Estimation:**

Request for Proposal

In Scope and Out of Scope

Different Testing Phases

Kind of Testing to be Performed

Work Streams

Estimated Number of User Story Points

Use Case Points to Test Cases Conversion

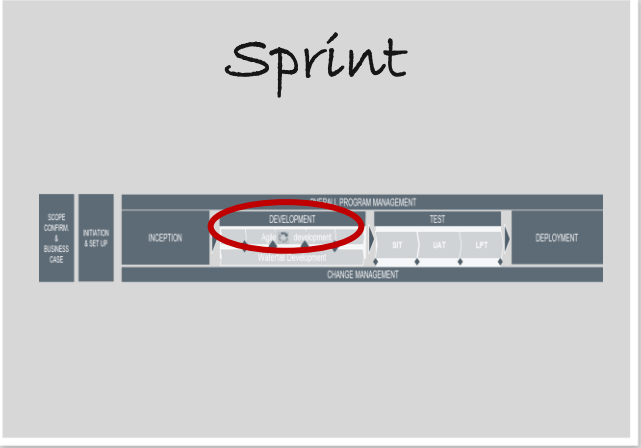
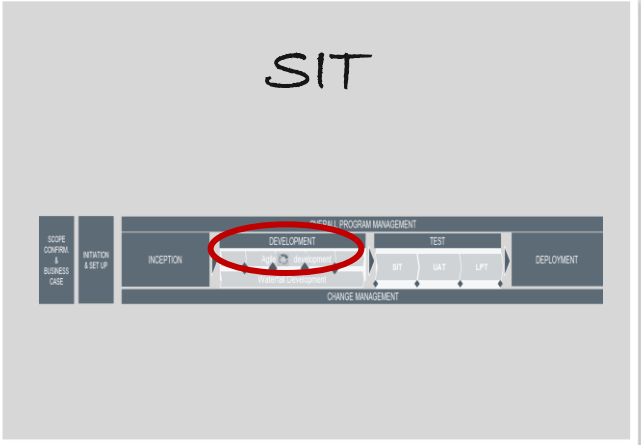
% Strategy for Test Cases Split up

Base Factors - Workstream wise



% of Test Cases to be automated

% of Test cases for Regression

# Entry and Exit criteria (1/2)

Phase	Description	Goal	When	Criteria
<p><i>Sprint</i></p> 	Definition of Done (DoD)	Validate the completion of the sprint	End of each Sprint	<ul style="list-style-type: none"> <li>Expected deliverables clearly defined, validated, and accepted by the nominated responsible</li> <li>All necessary Functional, Integration, Regression Tests done and passed</li> <li>All Acceptance criteria are met</li> <li>Required documentation prepared or updated</li> <li>Related test cases described and signed-off</li> <li>Less than 4 P4 defects, although moving to the next sprint would be possible if agree by the Program Directors</li> <li>Dependencies communicated to other teams</li> <li>Environment configuration instructions prepared</li> </ul>
<p><i>SIT</i></p> 	Entry SIT criteria	Validate the readiness to start SIT	End of Development Phase	<ul style="list-style-type: none"> <li>Exit criteria previous phase achieved</li> <li>Functionality available in the SIT environment</li> <li>SIT Test Plan created and signed off by business</li> <li>Test cases designed and signed off by business</li> <li>Environment readiness test successfully passed</li> </ul>
	Exit SIT criteria	Validate the completion the SIT	End of SIT Phase	<ul style="list-style-type: none"> <li>100% test cases SIT executed</li> <li>&gt;=95% test cases SIT passed</li> <li>0 critical/high defects</li> </ul>

# Entry and Exit criteria (2/2)

Phase	Description	Goal	When	Criteria
<p><b>UAT</b></p> 	Entry UAT criteria	Validate the readiness to start UAT	End of SIT E2E tests	<ul style="list-style-type: none"> <li>Exit criteria for SIT E2E tests achieved</li> <li>Functionality available in the UAT environment</li> <li>UAT Test Planning designed and signed off</li> <li>Test cases designed and signed off</li> <li>Environment readiness test successfully passed</li> </ul>
	Exit UAT criteria	Validate the completion the UAT	End of UAT	<ul style="list-style-type: none"> <li>100% test cases UAT executed</li> <li>&gt;= 95% test cases UAT passed</li> <li>0 critical/high defects</li> <li>Production environment ready</li> </ul>
<p><b>PROD</b></p> 	Entry production criteria	Validate the readiness for Deployment	End of UAT	<ul style="list-style-type: none"> <li>Expected deliverables clearly defined, validated and accepted by the nominated responsible</li> <li>Exit criteria of UAT is achieved</li> <li>Functionality available in the PROD environment</li> <li>Deployment Verification Test (DVT) successfully passed</li> <li>0 critical/high defects in DVT</li> </ul>

# Testing Measurements and Metrics (1/3)

## Data Collection:

- Total story points committed in Sprint
- Total story points completed in Sprint
- Functional Manual cases executed in SIT
- Functional Automation cases executed in SIT
- Regression Manual cases executed in SIT
- Regression Automation cases executed in SIT
- Total End to End cases executed in SIT

# Testing Measurements and Metrics (2/3)

## Data Collection:

- # Valid Defects found in Sprint
- # Invalid Defects found in Sprint
- # Valid Defects found by Functional Manual execution in SIT
- # Valid Defects found by Functional Automation execution in SIT
- # Valid Defects found in Regression in SIT
- Total Valid Defects found in SIT
- Total Invalid Defects found in SIT
- Total # of Sev1 Defects reported in Sprint
- Total # of Sev2 Defects reported in Sprint
- Total # of Sev1 Defects reported in SIT
- Total # of Sev2 Defects reported in SIT

# Testing Measurements and Metrics (3/3)

## Key Metrics:

- Error Discovery Rate (EDR) in Sprint
- Error Discovery Rate (EDR) in SIT
- Defect Density by Tested Story Points
- Automation Coverage %
- Automation Leverage %
- % of Total Defects found by Automation
- Defect Rejection %
- Sprint Velocity

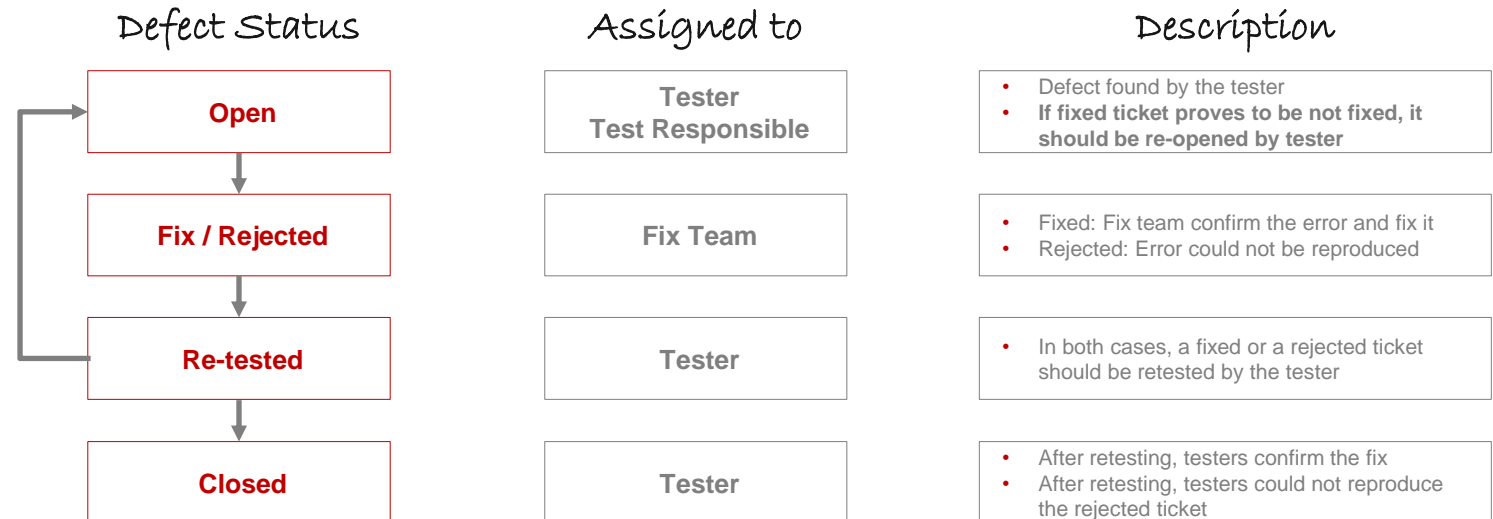


# Defect Management (1/2)

- Unexpected results discovered during testing should be recorded as a **Defect** with a reference to the test case mentioned
- Depending on the work item concerned the defect is assigned to a person who is in charge to analyse and provide a feedback to the tester
- Progress and updates on defects are discussed during test/defect status meetings
- Defects can be coordinated within the overall test management
- Each defect may include the following:

- Unique Defect ID
- Date
- Defect owner
- Assigned to
- Defect Status
- Priority
- Severity
- Description
- Test Case ID (for traceability)
- Area Impacted
- Resolution Date

*Defect Lifecycle*



# Defect Management (2/2)

- Defect report should include information on the number defects with their relevant status, priority and severity.

IMPACT MATRIX (DEFECTS)			
	SEVERITY		
PRIORITY	MODERATE/MINOR	MAJOR	CRITICAL
LOW	NOT URGENT	NORMAL	NORMAL
MEDIUM	NORMAL	URGENT	URGENT
HIGH	URGENT	URGENT	VERY URGENT
VERY HIGH	URGENT	VERY URGENT	CRITICAL
URGENT	VERY URGENT	CRITICAL	CRITICAL

SEVERITY	DESCRIPTION	RESOLUTION TIME
Critical	<ul style="list-style-type: none"> <li>At least one operational unit must be affected</li> <li>Using the required function is impossible or unsuitable for the users</li> <li>The fault must be related to a business-critical process</li> <li>There is no workaround for executing the required function</li> </ul>	TBD
High	<ul style="list-style-type: none"> <li>There is no business interrupting error</li> <li>The error has essential influence on the business processing, but does not prevent the business process from being completed or the error concerns a process that is not high priority or business-critical process is not available for a small number of users</li> </ul>	TBD
Medium / Low	<ul style="list-style-type: none"> <li>No business interrupting error and no error classified as high exist.</li> </ul>	TBD