An Organic Mechanism

Effective Localization QA in Agile Development Environments
Agenda

Part 1. Agile Principles in Localization
   1.1. Agile – The Origins
   1.2. Localization and Waterfall
   1.3. What is agile localization?
   1.4. Team workflow and ideology

Part 2. Agile Testing Preparation
   2.1. Scoping and pre-planning
   2.2. Test strategy and planning
   2.3. Optimizing localization processes

Part 3. Localization Testing
   3.1. Testing methodologies
   3.2. Testing areas
   3.3. Test cycles
PART I

Agile Principles in Localization
Localization as end-of-chain effort

Poor adaptation to late changes

Low cultural awareness in design stage

Testing often generalized and non-specific

Focus on activities and departments

LOCALIZATION IN WATERFALL
WHAT IS AGILE LOCALIZATION?

- Fast response to changing requirements
- Shorter deployment periods
- Accurate work tracking
- Test-driven development
- Collaborative endeavour
- Continuous project integration
AGILE EMANCIPATES TEAMS...

BUT miscommunication and unclarity compromise results.
AGILE EMANCIPATES TEAMS...

Emphasis on coordination and communication. More transparency. Continuous adaptation to change.
PART II

Agile Localization – Testing Preparation
Localization is at the crossroads of design, development and testing.

Localization is NOT a service “black box”. GLASS-BOX PRINCIPLE
TECHNICAL PRE-ASSESSMENT

- Configure environment
- Sync with code repositories
- Identify actors and integrate
  LSPs in workflow
- Resource planning and
  necessary outsourcings
- Release plan settlement
  (simultaneous or prioritized
  language deployment)
LOCALIZATION SCOPING

- Establish localization prioritization based on User Story prioritization and PM requirements
- Assess existing resources
- Ensure reusability through minimalism
- Refactor for language-neutral frameworks
TEST STRATEGY

- Regression testing
- User acceptance testing
- Prioritize and systematize:
  - Exploratory testing
  - Usability testing
  - Functionality testing
- Draft test cases
- Sync with development

PROJECT AND REQUIREMENT-SPECIFIC
**LOCALIZATION PLANNING - TRACEABILITY**

"We want to push users to jump to our website from our product."

"We need an icon with a direct link. The icon should be gender and culturally-neutral."

"The icon should be accompanied with the label "For more info, click [here]."

### Feature/Epic
- User experience guidelines
- Consistent look and feel
- Cultural assessment as required
- Define localization personas
- Draft localization concept

### User Story
- Localization as part of UI native design
- Draft mockups and optimize iteratively
- Leverage/refactor existing test
- Localization as part of acceptance criteria

### Test Cases
- Functional: drafted against the User Story/Feature (ex. functionality, linguistic)
- Non-functional: look and feel, UI consistency
TERMINOLOGY PREPARATION

• *Ad-hoc* extraction is recommended to avoid workflow blockers for non-critical terms

• Work on critical-domain Epics by domain (ex. product module names, specific user-oriented terms)

• Centralize terminology management

• Define product terminology at Epic/Feature level, *never at User Story level*
INTEGRATION TESTING MODEL

Complementary Localization/Development Cycles

Best for centralized and tight teams.

Software development and localization processes are complementary.

Translation is carried out within the iteration.

Reviews are performed Just-in-Time by localizers (functional cosmetic) and translators.
**MULTILINGUAL PROCESS**

**PREPARATION**
- Draft guidelines for master localization
- Mock-ups and text with Dev. & Design

**LOCALIZATION PROCESS**
- Master Localization review and testing
- Pseudo-translated builds

**PRE-MULTILINGUAL LOCALIZATION**
- Review & analysis
- Guidelines adjustment for other languages
USER STORY LOCALIZATION MATURATION

Single-sourcing your localization assets is key.

Priorize languages and start testing by stages.
ITERATIVE TESTING MODEL

- GENERAL TESTING
- INTERNATIONALIZATION TESTING
- LOCALIZATION TESTING

Project Start → Sprint 2 → Sprint 3 → Sprint 4 → Sprint 5 → Sprint 6 → Sprint 7 - Release
PART III

Agile Localization - Testing
REVVING UP

- Continuous communication
- Implement centralized tools
- Clarify workflows
- Sync and review materials

Indetermination leads to poor efficiency

ENSURE YOUR TEAM KNOWS WHAT TO DO
TESTING METHODOLOGIES

- Integrated test and development environments
- Combination of testing methods
- Resource and project-dependent
- Accurate test case management
TESTING METHODOLOGIES

1. MANUAL
2. SEMI-AUTOMATED
3. AUTOMATED

Advantages:

- Language consistency
- Style/register issues
- Terminology issues
- Feedback on in-context usability testing
TESTING METHODOLOGIES

1. MANUAL
2. SEMI-AUTOMATED
3. AUTOMATED

Disadvantages:

- Training curve required
- High risk of inconsistencies
- Time-consuming
- Harder logistical management
TESTING METHODOLOGIES

1. MANUAL
2. SEMI-AUTOMATED
3. AUTOMATED

Human factor with tool support
Test case automatization
Suitable for text checks
Excellent in GUI testing
TESTING METHODOLOGIES

1. MANUAL
2. SEMI-AUTOMATED
3. AUTOMATED

Functional tests
Unit tests
GUI validation
Reusability
GUI prototype validation
LOCALIZATION TESTING

1. LINGUISTIC
   - CAPITALIZATION
   - CULTURAL ADEQUACY
   - SPELLCHECKING
   - STYLE/REGISTER
   - TERMINOLOGY
LOCALIZATION TESTING

1. LINGUISTIC
   - Capitalization
   - Cultural Adequacy
   - Spellchecking
   - Style/Register
   - Terminology

2. FUNCTIONALITY
   - Installation
   - Menus and Hotkeys
   - Replicability in All Languages
   - Input Correctness
   - Compatibility
LOCALIZATION TESTING

1. LINGUISTIC
   - Capitalization
   - Cultural Adequacy
   - Spellchecking
   - Style/Register
   - Terminology

2. FUNCTIONALITY
   - Installation
   - Menus and Hotkeys
   - Replicability in all Languages
   - Input Correctness
   - Compatibility

3. COSMETIC
   - Truncation
   - Content
   - GUI Integration
   - Consistency
   - Layout
AUTOMATED TESTING

- Test cases replicability
- GUI validation:
  - Alignment, truncation
  - Overlapping
- Text:
  - Rendering
  - Blacklists, hotkeys
  - Linguistic checks
LOCALIZATION UNIT TESTS

Ideal in test-driven development
Automation on code level
Limit regressions
Identify bugs sooner
Reduce testing effort
Improve code coverage

@implementation tests
-(void)testCreateFoo {
 STAssertNotNil (foo, @"Unable to create dialog");
}
...
LOCALIZATION TEST STAGES

Localization requirements
Technical framework
Product internationalization
Cultural appropriateness

ACCEPTANCE
String call semantics
Strings ID and content
Locale files dependencies
Time zone/cultural marks
GUI layout (mobile and desktop apps)

INTEGRATION
Rendering engine
GUI changes
Frameworks
Test cases

TEST CYCLE

UNIT TESTS

REGRESSION
Bug fixing
Text refactoring
Updated icons and graphics
Code updates
SUMMARY

• Integrate localization in release and iteration planning
• Improve traceability
• Centralize tools and processes
• Localization QA as integral part of the testing plan
• Improve automation
• Integrate all testing actors in one centralized workflow
QUESTIONS? DOUBTS?

THANK YOU FOR YOUR ATTENTION.

ALBERTO FERREIRA
ALBERTO.FERREIRA@AVIRA.COM
AVIRA OPERATIONS GMBH & CO. KG
RECOMMENDED READINGS


