WIRELESS INTERNET SOFTWARE ENGINEERING IST-2000-30028

Title:
Service Engineering Process
(The reference process model)

Version: 03.05
Date : 17 Sep 04
Pages : 76

Author(s):
Alexis Ocampo, Juergen Muench

To:
WISE CONSORTIUM

The WISE Consortium consists of:

Investnet, Motorola Technology Center Italy, Sodalia s.p.A, Sonera, Solid EMEA North, Fraunhofer IESE, Politecnico di Torino, VTT Electronics

Printed on:
17/09/2004 10.22

Status:

[ ] Draft
[ ] To be reviewed
[ ] Proposal
[X] Final/Released

Confidentiality:

[X] Public - Intended for public use
[ ] Restricted - Intended for WISE consortium only
[ ] Confidential - Intended for individual partner only

Deliverable ID: D2 (Part B)

Title:

Service Engineering Process (The reference process model)

Summary / Contents:
This document is part of deliverable D2, which describes the work done and results obtained for the WP1-Task: “Define a process to engineer services” of the WISE project. Deliverable D2 includes three parts: Part A: Service Engineering Process (Empirical approach for creating the reference process model), Part B: Service Engineering Process (The reference process model), and Part C: Service Engineering Process (Pilot Processes).
This part of the deliverable presents the reference process model: The first chapter describes the motivation of the work. As part of the strategy, the survey of related work and its results are presented in the second chapter. Chapter 2 presents the reference process model as a process handbook with complete descriptions of processes, artifacts, roles and tools, product flows, and process views.

TABLE OF CONTENTS

CHANGE LOG

Vers.	Date	Author	Description
01.02	05/04/02	U. Becker-Kornstaedt, A Ocampo	Final version D2.V01.02
01.03	16/10/02	A. Ocampo, Dr. Jürgen Muench	Draft version D2.V01.03
01.03	12/11/02	A. Ocampo, Dr. Jürgen Muench	Final version D2.V01.03
01.04	08/01/03	A. Ocampo, Dr. Jürgen Muench	Final version D2.V01.04
02.01	06/04/03	A. Ocampo, Dr. Jürgen Muench	Draft version D2.V02.01
02.01	30/04/03	A. Ocampo, Dr. Jürgen Muench	Final version D2.V02.02
03.01	06/10/03	A. Ocampo, Dr. Jürgen Muench	Draft version D2.V03.01
03.02	23/10/03	A. Ocampo, Dr. Jürgen Muench	Final version D2.V03.02
05.01	23/06/04	A. Ocampo, Dr. Jürgen Muench	Draft version D2.V05.01

APPLICABLE DOCUMENT LIST

Ref.	Title, author, source, date, status	Identification
1	Indicators; Dr. Jürgen Münch; Fraunhofer IESE; 10/06/2003; Proposal	D8-V2
2	Analysis: Fabio Bella, Dr. Jürgen Münch, Alexis Ocampo; 09/06/2003	D9-V2
3	Heterogeneous Clients; Filippo Forchinno, Mario Negro Ponzi; 08/09/2003	D3-V2

1. INTRODUCTION

1.1 MOTIVATION

The lack of knowledge about wireless technologies, the unavoidable growth of this type of applications in the coming years, and the need for a systematic approach to developing these applications are important reasons to justify the creation of such a reference process. This process has been descriptively elicited in a systematic way through the development of pilot projects and literature study.

We call the resulting comprehensive process model in this report reference process, because is intended to be used as a reference for developers and managers providing initial orientation in selecting appropriate technologies.
Creating a reference process solely from observing software projects limits the level of detail and precision to the abilities of the respective project organizations in which the projects were performed. Therefore, we enhanced the developed reference process carefully with knowledge from other sources (e.g., literature, experience reports from similar projects).
The reference process model presented in this section does not significantly differ from traditional iterative process models on the life cycle level, but it includes domain-specific and experience-based guidance on the level of engineering processes. At the moment, it is difficult to provide more concrete decision support for project planning due to inexperience and to the recentness of the domain, but recording experience is considered to be part of the future work in order to understand the complexity of the domain.

1.2 DOCUMENTATION

The deliverable (D2.V5) documents the methodology to create the reference process model. Part A presents the results of the step “Survey processes for software and system engineering”, “Define new process to develop wireless services”, “Define measures and indicators”, and “evaluate quality related activities”. Part B updates the reference process model as a result from the step “Define new process to develop wireless services, in terms of activities, techniques, tools, deliverables, and milestones“. Part C documents the pilots’ process models used for iteration 3 as a result of the step “Elicit existing process knowledge”. Please note in Table 1 that the documents’ version differs from the pilot’s process models version (PVX) and the Reference Process Model version (WISEPVX). This document corresponds to part B of the deliverable D2.V5.

Table 1. Documented results WP1 – Task 1.2

Process Models\Iteration	Start iteration 1	End iteration 1	Start iteration 2	End iteration 2	Start iteration 3	End iteration 3
Pilots 1 and pilot 2 planned processes	D2.V0 (PV0)		D2.V2 (PV1)		D2.V4 (PV2)	
Empirical approach for creating the Reference Process Model						D2.V5.A
Reference Process Model		D2.V1 (WISEPV1)	D2.V2 (WISEPV1)	D2.V3 (WISEPV2)	D2.V4 (WISEPV2)	**D2.V5.B (WISEPV3)**
Pilot 1 and pilot 2 actual processes		D2.V1 (PV1)		D2.V3 (PV2)		D2.V5.C (PV3)

2. REFERENCE PROCESS MODEL: WISEP

2.1 OVERVIEW OF THE PROCESS MODEL

2.1.1 Activities

The following activities are present in this process model:

- Requirements Phase
  - Select Requirements
    - Analyze UI Feasibility
    - Write Scenarios
    - Write Requests
    - Select Feasible Requests
  - Study Feasibility
    - Search Possible Solutions
    - Test Possible Solutions
  - Specify Requirements

- Design Phase
  - Design High Level
    - Specify Conceptual Design
    - Aquire COTS
      - Identify COTS Products
      - Evaluate COTS Products
      - Select COTS Products
  - Design Low Level
  - Inspect Design

- Coding Phase
  - Code
  - Test Units
  - Test Integration
  - Release Code

- Plan Testing Phase
  - Plan Tests
  - Build Test Framework

- Testing Phase
  - Test System
  - Test Acceptance
  - Test Usability
  - Analyze Defect

2.1.2 Artifacts

The following artifacts are present in this process model:

- Requests From Customer
- Requirements Specification
- List Possible Solutions
- Feasibility Study

- WISA Architectural Guidelines
- Architecture Document
- Candidate COTS Products
- Evaluation Results
- Selected COTS Products
- Inspection Log
- Design Inspection Checklist
- Source Code
- Integration Report
- Integrated Code
- Test Plan
- Test Framework
- Product Tested
- Test Report
- Usability Interview
- Usability Report
- Acceptance Test Report
- Analyze Defect Report

2.1.3 Roles

The following roles are present in this process model:

- Customer
- Developer
- Project Leader
- Project Manager
- Technical Leader

2.1.4 Tools

The following tools are present in this process model:

- Integrated Development Environment
- Real Mobile Device
- Text Editor
- UML Editor

2.1.5 Process View

The following is a global view for this process model:

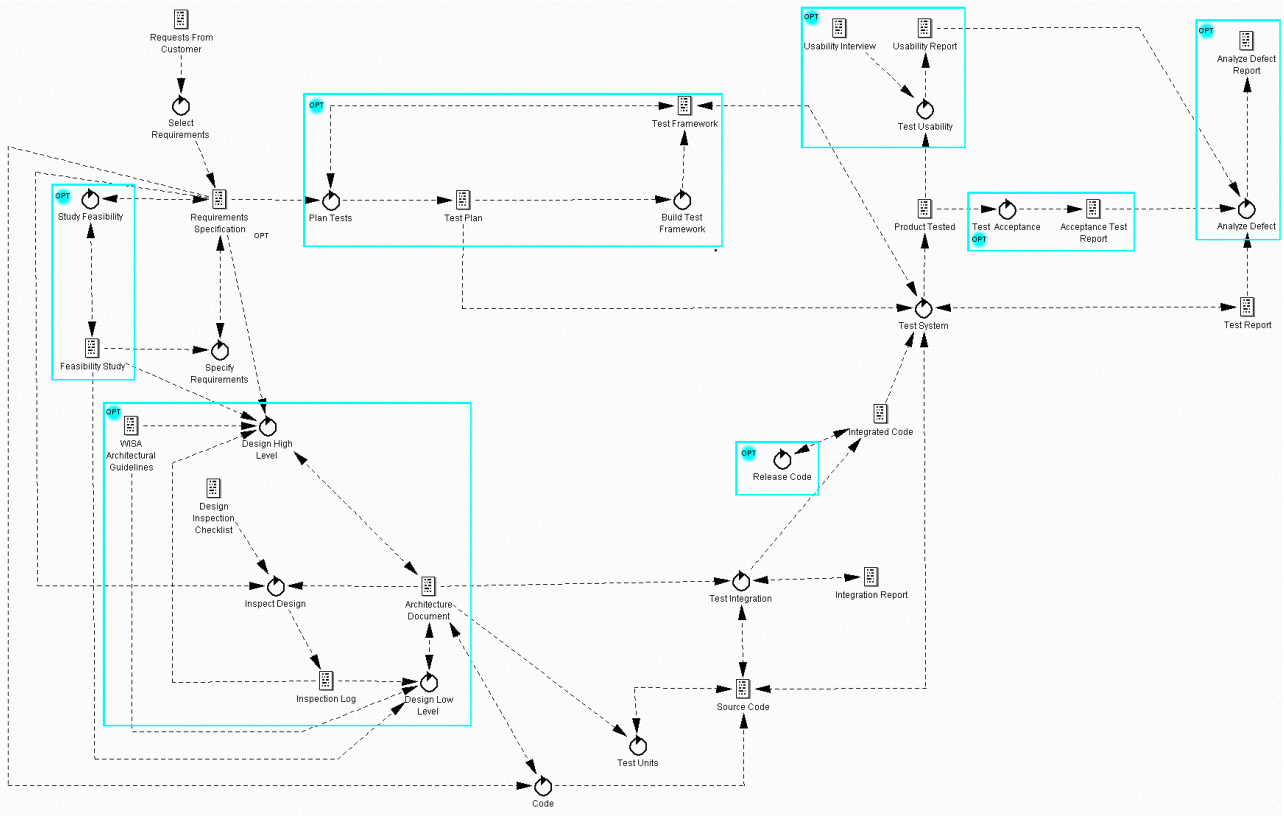

Figure 2-1. Global view.

2.2 WISEP /PHASE_OVERVIEW

The following phases are described in this process description:

- Requirements Phase.
- Design Phase.
- Coding Phase.
- Plan Testing Phase.
- Testing Phase.

The following graph depicts the product flow between the different phases.

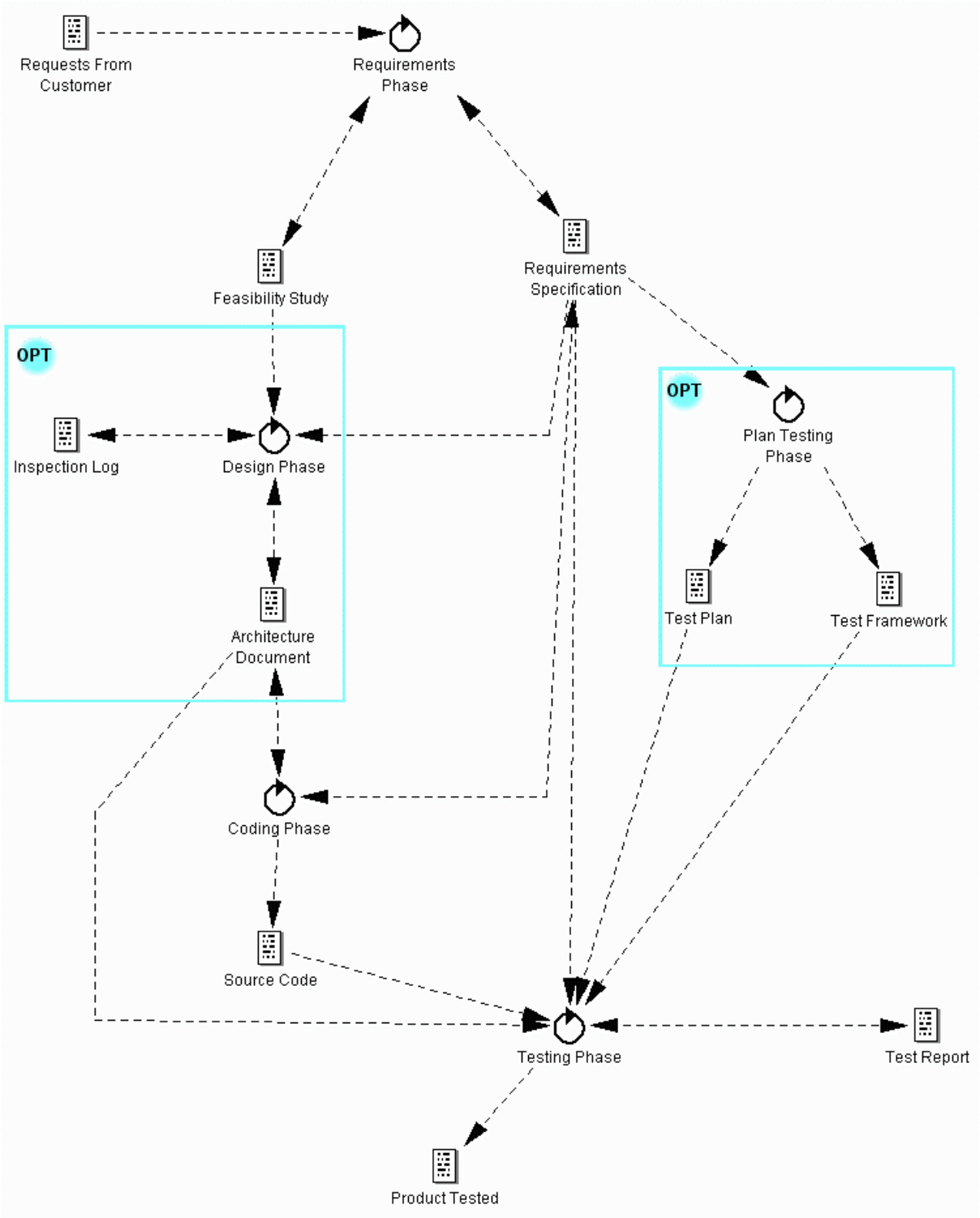

Figure 2-2. Product flow between phases.

2.3 WISEP / REQUIREMENTS PHASE.

2.3.1 Purpose

To produce or obtain clear and unambiguous requirements.

2.3.2 Type

Common

2.3.3 Description

1.Select requirements.
2. Perform a feasibility study.
3. Specify the requirements.

2.3.4 Risk Factors

Technological risks
- Diversity of wireless network standards
- Lack of experience with new charging and billing models
- Small screens
- Limited keyboards
- Diversity of input mechanisms
- Limited device power capability
- Limited non-volatile memory storage

Risk mitigation hints:
Hint 1
Understand the set of available technologies. [1] presents a survey with a classification of available technologies, and their relationships with device independency in the context of wireless Internet services.
Hint 2
Read the device independency principles document [2]. At the moment, the principles are general, but they will be specialized with guidelines and requirements to obtain device independency as well as to concentrate all standardization efforts in one place.
Hint 3
Follow the recomendations given by the Heterogeneous Client document from the WISE project [3]. Here a list of recomendations for thin clients and fat clients is given.
Organizational risks
- Ambiguous, incomplete, or inconsistent requests.
- Lack of experience planning application releases in the wireless domain.

Risk mitigation hints:
Hint 1
Involve actively the customer through interviews, meetings, and discussions in order to clarify his requests and priorities.
Hint 2
Understand the problem of planning releases by looking at studies in the requirements engineering field. e.g., [4]. One important statement is the definition of the release planning activity as a wicked problem. A wicked problem is a problem that stops when there is no more time, no money or the solution is good enough. It is a problem with no optimal solution, unique and irrepeteable, therefore no measures of success are possible.
Hint 3
Extreme programming [5] proposes user stories as a medium to capture functional requirements in a simple, non-formal language. Hint 4
Take some ideas from the adaptive software development (ASD) [6]. It proposes to face uncertainty with short delivery iterations; new requirements and technical information with intensive collaboration among

managers, customers and developers; and process improvement with reviews after each iteration and project retrospectives
Hint 5
Take some ideas from the spiral model [7]. It assumes risks as the driving force of software projects. This model proposes ongoing refinement of the system specification through cycles, and each cycle is risk assessed. A risk assessment determines if a project continues or is cancelled.
Hint 6
Take some ideas from Construction Planning [8]. It is a nine-step process used for the development of radio systems, which allows project managers to model and plan the functionality of increments, track their evolution, and update the project plan. The method uses as basis good and bad increments planning experiences from real projects. Construction Planning helps to have control of increments, and receive constant feedback from the customer on the quality of the products.
A planned release or increment also determines which customer will get special features and what will be the quality in a given point of time. This approach helps to have control of increments, and receive constant feedback from the customer on the quality of the products.

2.3.5 Input Criteria

1. The collected requirements from the customer were identified clearly as a new development from scratch or a modification of an existing product.
2. A defect from a previous version was identified and documented. The defect was traced back to an specified requirement. The requirement must be reworked.

2.3.6 Exit Criteria

1. A completed requirements specification.
2. The results of the feasibility study.

2.3.7 Product Flow

This activity consumes the following artifacts:
- Requests From Customer

This activity modifies the following artifacts:
- Requirements Specification
- Feasibility Study

This activity does not produce any artifacts.

2.3.8 Involved Roles

The following roles are involved with this activity:
- Customer
- Developer
- Project Leader

2.3.9 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device
- Text Editor

2.3.10 Subactivities

The following are the subactivities of this activity:
- Select Requirements
- Study Feasibility
- Specify Requirements

2.3.11 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity. You can click on them to get a full size view:

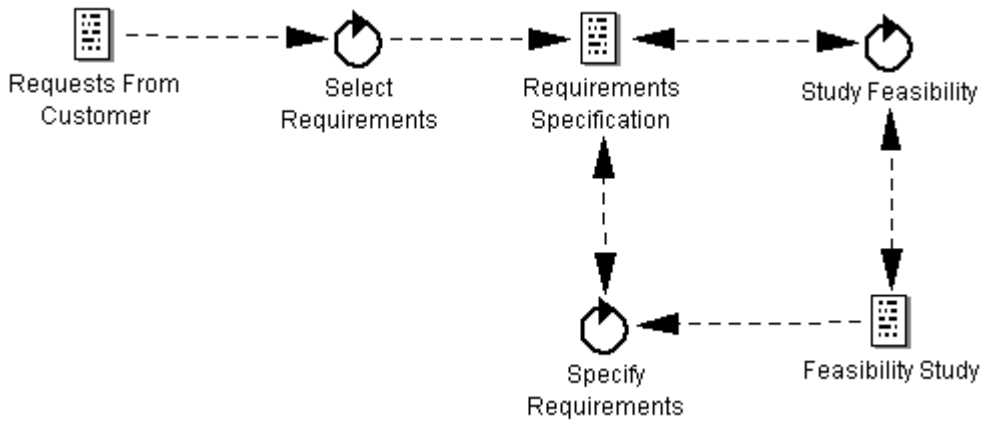

Figure 2-3. Product flow requirements phase.

2.3.12 Process View: *Roles*

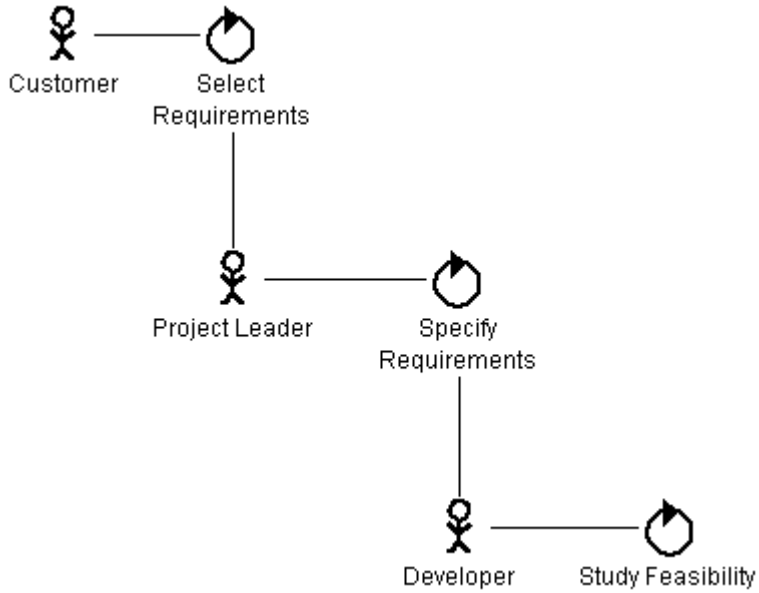

Figure 2-4. Process view: requirements phase.

2.3.13 Process View: *Tools*

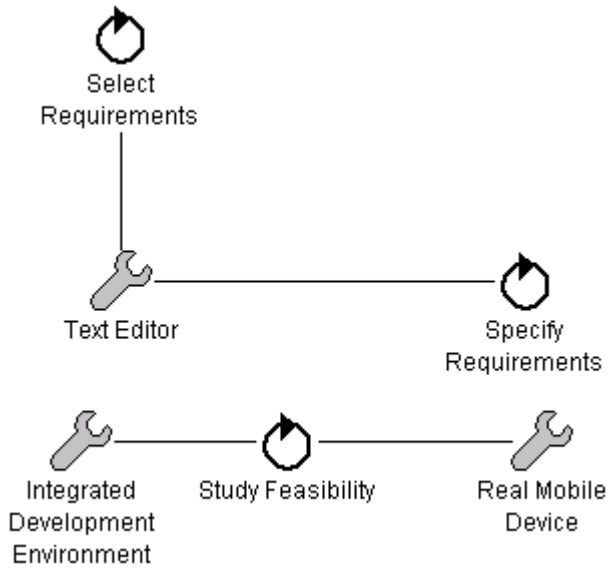

Figure 2-5. Process view: requirements phase tools

2.3.14 Activity: *Select Requirements*

2.3.14.1 Purpose

To analyze, clarify, and state, the set of requirements to be implemented.

2.3.14.2 Description

Alternative 1
1. Select feasible request
2. Write requests
Alternative 2
1. Analyze User Interface Feasibility
2. Write scenarios

2.3.14.3 Type

Common

2.3.14.4 Input Criteria

1. The collected requirements from the customer were identified clearly as a new development from scratch or a modification of an existing product.
2. A defect from a previous version was identified and documented. The defect was traced back to an specific requirement. The requirement must be reworked.

2.3.14.5 Exit Criteria

1. The first draft of the requirements document is provided.
2. The information was approved by the customer.

2.3.14.6 Product Flow

This activity consumes the following artifacts:
- Requests From Customer

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Requirements Specification

2.3.14.7 Involved Roles

The following roles are involved in this activity:
- Customer
- Project Leader

2.3.14.8 Used Tools

This activity uses the following tools:
- Text Editor

2.3.14.9 Subactivities

The following are the subactivities of this activity:
- Analyze UI Feasibility
- Write Scenarios
- Write Requests
- Select Feasible Requests

2.3.14.10 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity.

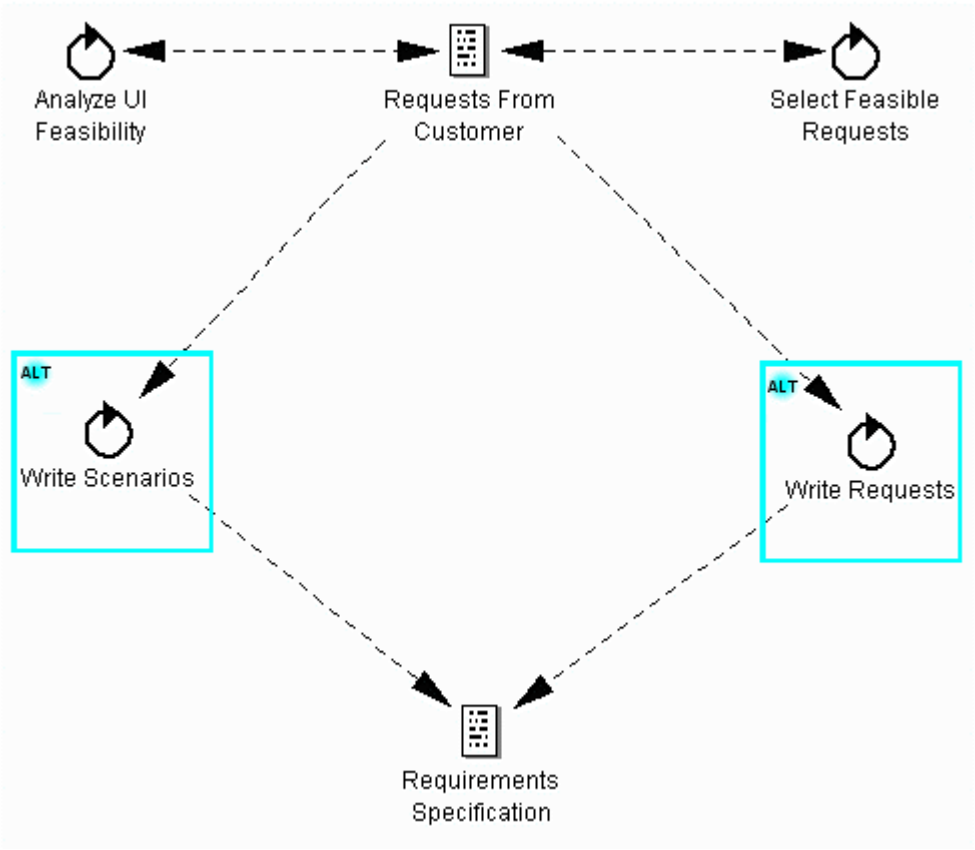

Figure 2-6. Product flow: Select Requirements.

2.3.14.11 Process View*: Roles*

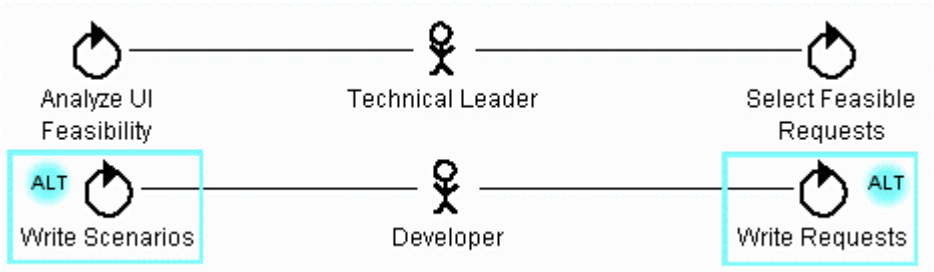

Figure 2-7. Process view: Select Requirements Roles

2.3.14.12 Process View: *Tools*

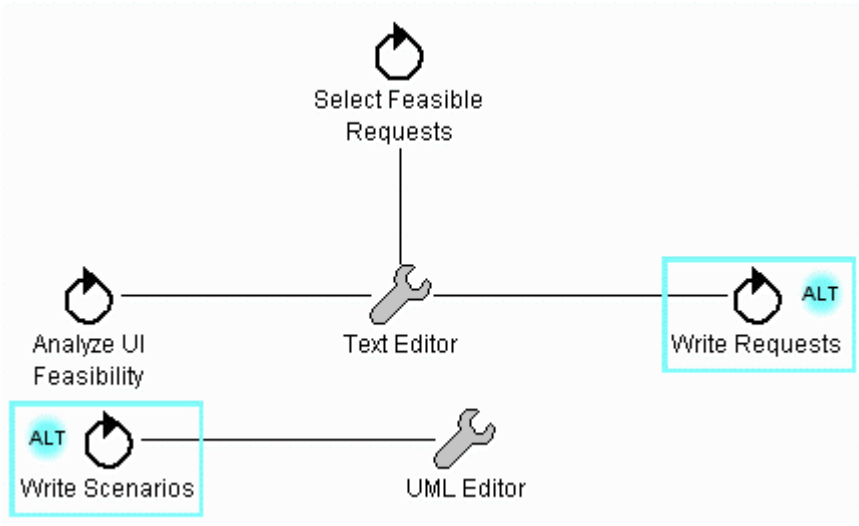

Figure 2-8. Process view: Select Requirements Tools.

2.3.14.13 Activity: *Analyze UI Feasibility*

2.3.14.13.1 Purpose

1. To analyze carefully if the requests from customer concerning the user interface are possible to be implemented.

2.3.14.13.2 Description

1. Validate the requests of the customer with the existent capabilities, everytime a new description of the service arrives.
2. Once this analysis is done, decide whether accepting the user interface requirements or rejecting them. Sometimes, developers are not sure about the feasibility of requirements. These requirements can be partially received but research must be done deeply during the feasibility study.

2.3.14.13.3 Type

Common

2.3.14.13.4 Input Criteria

1. The collected requirements from the customer were identified clearly as a new development from scratch or a modification of an existing product.
2. A defect from a previous version was identified and documented. The defect was traced back to an specific requirement. The requirement must be reworked.

2.3.14.13.5 Exit Criteria

1. Updated requests from customer with the list of those that can be implemented and those that are partially accepted but will be subject of a feasibility study.

2.3.14.13.6 Product Flow

This activity does not consume any artifacts.
This activity modifies the following artifacts:

- Requests From Customer

This activity does not produce any artifacts.

2.3.14.13.7 Involved Roles

The following roles are involved with this activity:

- Technical Leader

2.3.14.13.8 Used Tools

This activity uses the following tools:

- Text Editor

2.3.14.14 Activity: *Write Scenarios*

2.3.14.14.1 Purpose

1. To specify as scenarios (use cases) the set of request from customer to be implemented.

2.3.14.14.2 Description

1. Go through the customer requests, understand them, clarify ambiguities, and identifying possible scenarios. A scenario is represented as an UML use case diagram. An scenario has roles and use cases. The scenarios are collected, spitted, and distributed among developers.
2. Specify the scenarios for the first time as requirements in the requirements specification document.

2.3.14.14.3 Type

Alternative

2.3.14.14.4 Input Criteria

1. Updated requests from customer with the list of those that can be implemented and those that are partially accepted but will be subject of a feasibility study.

2.3.14.14.5 Exit Criteria

1.Scenarios specified in the requirements document.

2.3.14.14.6 Product Flow

This activity consumes the following artifacts:

- Requests From Customer

This activity does not modify any artifacts.
This activity produces the following artifacts:

- Requirements Specification

2.3.14.14.7 Involved Roles

The following roles are involved with this activity:

- Developer

2.3.14.14.8 Used Tools

This activity uses the following tools:

- UML Editor

2.3.14.15 Activity: *Write Requests*

2.3.14.15.1 Purpose

To write down in any form whether plain text or by using special templates the requests from customer to be implemented.

2.3.14.15.2 Description

1. Write down in informal text the set of requests from customers that were approved to be implemented (Including the partially accepted).

2.3.14.15.3 Type

Alternative

2.3.14.15.4 Input Criteria

1. Updated requests from customer with the list of those that can be implemented and those that are partially accepted but will be subject of a feasibility study.

2.3.14.15.5 Exit Criteria

1.Requests specified in the requirements document.

2.3.14.15.6 Product Flow

This activity consumes the following artifacts:

- Requests From Customer

This activity does not modify any artifacts.
This activity produces the following artifacts:

- Requirements Specification

2.3.14.15.7 Involved Roles

The following roles are involved with this activity:

- Developer

2.3.14.15.8 Used Tools

This activity uses the following tools:

- Text Editor

2.3.14.16 Activity: *Select Feasible Requests*

2.3.14.16.1 Purpose

To define the set of requirements to be implemented, and the set of requirements to be consider for a feasibility study.

2.3.14.16.2 Description

1. Perform a meeting to establish what could be implemented and what should be carried out as a feasibility study.
2. Update the requests from customer with the outputs of the discussion.

2.3.14.16.3 Type

Alternative

2.3.14.16.4 Input Criteria

1. The collected requirements from the customer were identified clearly as a new development from scratch or a modification of an existing product.
2. A defect from a previous version was identified and documented. The defect was traced back to an specific requirement. The requirement must be reworked.

2.3.14.16.5 Exit Criteria

1. Updated requests from customer with the list of those that can be implemented and those that are partially accepted but will be subject of a feasibility study.

2.3.14.16.6 Product Flow

This activity does not consume any artifacts.
This activity modifies the following artifacts:

- Requests From Customer

This activity does not produce any artifacts

2.3.14.16.7 Involved Roles

The following roles are involved with this activity:

- Technical Leader

2.3.14.16.8 Used Tools

This activity uses the following tools:

- Text Editor

2.3.15 Activity: *Study Feasibility*

2.3.15.1 Purpose

To resolve technological doubts found in the requirements specification and provide sound basis for accepting or rejecting these requirements.

2.3.15.2 Description

The feasibility study is often carried out on the following limitations; Heterogeneous clients, network issues, internal and external connection requirements.
1. Search after possible solutions.
2. Test possible solutions.

2.3.15.3 Type

Common

2.3.15.4 Input Criteria

1. The requirements specification document approved by the customer.

2.3.15.5 Exit Criteria

1. A complete feasability study that comprises guidelines, which solve technological doubts.

2.3.15.6 Product Flow

This activity does not consume any artifacts.
This activity modifies the following artifacts:

- Requirements Specification

- Feasibility Study

This activity does not produce any artifacts.

2.3.15.7 Involved Roles

The following roles are involved in this activity:

- Developer

2.3.15.8 Used Tools

This activity uses the following tools:

- Integrated Development Environment
- Real Mobile Device

2.3.15.9 Subactivities

The following are the subactivities of this activity:

- Search Possible Solutions
- Test Possible Solutions

2.3.15.10 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity.

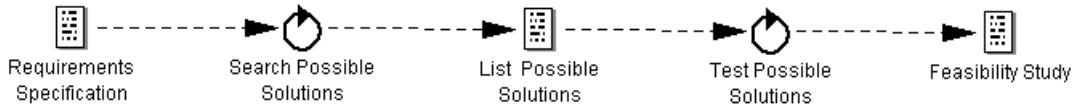

Figure 2-9. Product Flow: Study Feasibility

2.3.15.11 Process View*: Roles*

Figure 2-10. Process view. Study Feasibility Roles,

2.3.15.12 Process View*: Tool*

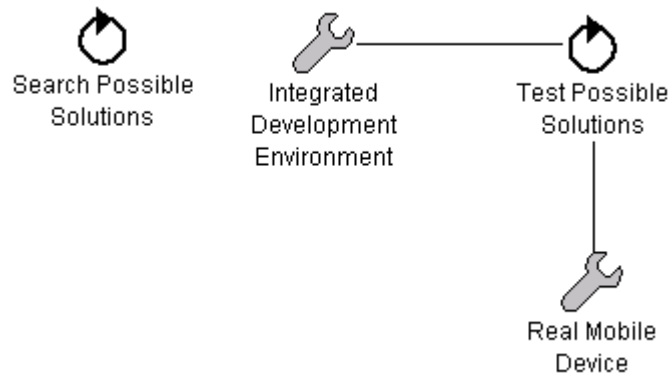

Figure 2-11. Process view: Study Feasibility Tools.

2.3.15.13 Activity: *Search Possible Solutions*

2.3.15.13.1 Purpose

To look after as many alternatives as possible to resolve technical doubts concerning the requirements specification.

2.3.15.13.2 Description

1. Interview experts on wireless technologies
2. Search in the Intranet or Internet.

2.3.15.13.3 Type

Common

2.3.15.13.4 Input Criteria

1. The first draft of the requirements document is provided.
2. The information was approved by the customer.

2.3.15.13.5 Exit Criteria

1. A completed list of possible solutions to technical doubts found in the requirements specification.

2.3.15.13.6 Product Flow

This activity consumes the following artifacts:
- Requirements Specification

This activity does not modify any artifacts.
This activity produces the following artifacts:
- List Possible Solutions

2.3.15.13.7 Involved Roles

The following roles are involved with this activity:
- Developer

2.3.15.13.8 Used Tools

This activity uses the following tools:
This activity uses no tools.

2.3.15.14 Activity: *Test Possible Solutions*

2.3.15.14.1 Purpose

To implement small prototypes in order to test technical doubts detected in the requirements specification.

2.3.15.14.2 Description

1. Design a small prototype for testing a possible solution.
2. Design test cases in order to validate the prototype.
3. Implement the prototype if needed.
4. Test the prototype.
5. Review the results.
6. Document the results in the Feasibility Study document.

7. Provide final conclusions on the feasibility study document.

2.3.15.14.3 Type

Common

2.3.15.14.4 Input Criteria

1.A completed list of possible solutions to technical doubts found in the requirements specification.

2.3.15.14.5 Exit Criteria

1. A complete feasibility study that comprises guidelines which solve technological obstacles.

2.3.15.14.6 Product Flow

This activity consumes the following artifacts:
- List Possible Solutions

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Feasibility Study

2.3.15.14.7 Involved Roles

The following roles are involved with this activity:
- Developer

2.3.15.14.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.3.16 Activity: *Specify Requirements*

2.3.16.1 Purpose

To specify the set of requirements to be implemented.

2.3.16.2 Description

1. Refine the requirements specification and produce a stable and complete version.
2. Inspect the requirements specification.
3. Ensure the specification is clear and unambiguous.
4. Resolve any questions.

2.3.16.3 Type

Common

2.3.16.4 Input Criteria

1. The initial requirements specification approved by the customer.
2. The results of the feasibility study.
3. A defect from a previous version was identified and documented. The defect was traced back to an specific requirement. The requirement must be reworked.

2.3.16.5 Exit Criteria

1. A completed requirements specification.

2.3.16.6 Product Flow

This activity consumes the following artifacts:
- Feasibility Study

This activity modifies the following artifacts:
- Requirements Specification

This activity does not produce any artifacts.

2.3.16.7 Involved Roles

The following roles are involved in this activity:
- Developer
- Project Leader

2.3.16.8 Used Tools

This activity uses the following tools:
- Text Editor

2.4 *WISEP* / *DESIGN PHASE*

2.4.1 Purpose

To produce high level and low level designs that meet the requirements specification.

2.4.2 Type

Optional

2.4.3 Description

1. A high - level or conceptual design is produced.
2. A low - level or concrete design is produced.
3. (Optional) An inspection of the design document is executed.

2.4.4 Risk Factors

Technological risks
- Diversity of wireless network standards
- Lack of experience with new charging and billing models
- Small screens
- Limited keyboards
- Diversity of input mechanisms
- Limited device power capability
- Limited non-volatile memory storage

Risk mitigation hints
Hint 1
Regarding high level design the use of patterns like the MODEL-VIEW-CONTROLLER. It is recommended for use in wireless Internet service applications by [9], where the logic is concentrated on the server and none or a minimum of the business logic is revealed on the client side. The use of this pattern can have additional benefits such as: All the components are defined logically, each component has a function, interfaces are defined between components, each component can be implemented as another pattern, high reusability, high flexibility, reduced cost, and higher quality.
Hint 2
Use flow charts to sketch the decisions to reach certain functionality.
Hint 3

Storyboards can be used to contain the sequence of the pages that a user will encounter within a site. These techniques were found to be used by many companies in order to design the navigation of the structure in a web site [10].

Hint 4

Take some ideas from Giannetti's [11] Device Independence Web Application Framework (DIWAF). The framework is based on the single authoring principle, which consists of designing for the most capable device and automatically adapting content to different device classes. Content, layout and style are separated for reuse whenever possible.

Hint 5

Take some ideas from scalaweb, which is a technique presented by [12] that allows authors to build a device-independent presentation model at design time. ScalableWeb is also based on the single authoring technique, where authors can produce the layout specification for the largest screen size of a given device, and then a rendering system renders the device presentation model into device-specific presentations [13].

Hint 6

Take some ideas from Mori et al.'s approach [14], which present an XML-based approach oriented to design applications that are device independent.

Hint 7

Regarding user interface design layout standards for using video, animation, graphics, colors, and navigational standards such as where to place buttons, and the use of scrollbars and menus.

Hint 8

Regarding scalability, understand the scalability design process introduced by [15] The scalability design process is based on a set of strategies useful when designing scalable Internet sites. The strategies are based on the design principles of a scalable architecture: divide and conquer, asynchrony, encapsulation, concurrency, and parsimony. The message is clear. Successful wireless Internet services need to be scalable, but scalability demands a detailed architecture, a detailed design, and, once implemented, requires monitoring and maintenance.

Hint 9

Use the techniques proposed by Friday et al. [16] that can be used to adapt the system and improve the quality of service of the network (QoS) at different levels (i.e., user, application, middleware, and transport). For example, the system can allow the user to change from synchronous to asynchronous tasks (user level), or through proxy services the application can use local substitute services based on cache information (application level). At the middleware level the information can be fetched only when needed (on demand), and finally at the transport level, data can be prioritized, reordered, and exchanged according to the bandwith situation.

Hint 10

Read the discussion of billing infrastructure and charging models for the actual and future Internet presented in [9].

Hint 11

Regarding charging and billing models, take some ideas from the Paris-Metro charging model proposed by [17]. This model supposes that the subscriber defines a travel class as an association between cost and network traffic. For example, the subscriber could define that he will use the network in first class or second class according to the association network traffic-cost. The network could also detect that the first class is full (i.e., high traffic), therefore all the subscribers who want to use the network will have to use only the second class. If the subscriber would like to use the first class for a given service, then he will have to pay the correspondent penalty. According to [18] this model introduces complexity to the network behavior, overhead to the subscriber, and what is most important for developers, changes to the software application and extensions to the communication protocols. Therefore, developers should ask themselves during the conception of the application's design how much the model of charging and billing impacts the system's architecture.

Hint 12

Regarding usability, take a look at Nerurkar [19] that mergs the GUI methodologies used for designing traditional systems with the new Web design techniques [20], in order to improve Web design methodologies. Nerurkar defends the fact that the essentials of user-centered interface can be applied for Web interface design.

Hint 13
Search and use specific guidelines for designing user interfaces for devices or families of devices. The big mobile device producers or programming platform providers offer them, for example, the MIDP style guide offered by Sun Microsystems, Inc [21].
Hint 14
Design layout standards for using video, animation, graphics, colors, and navigational standards such as where to place buttons, and the use of scrollbars and menus.

2.4.5 Input Criteria

1. The requirements document approved by the customer.
2. The results of the feasibility study.
3. A defect from a previous version was identified and documented . The defect was traced back to the design artifact. The design must be reworked.

2.4.6 Exit Criteria

1. A complete design of the conceptual and concrete levels of the architecture.

2.4.7 Product Flow

This activity consumes the following artifacts:
- Requirements Specification
- Feasibility Study

This activity modifies the following artifacts:
- Architecture Document
- Inspection Log

This activity does not produce any artifacts.

2.4.8 Subactivities

The following are the subactivities of this activity:
- Design High Level
- Design Low Level
- Inspect Design

2.4.9 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity. You can click on them to get a full size view:

- design_phase

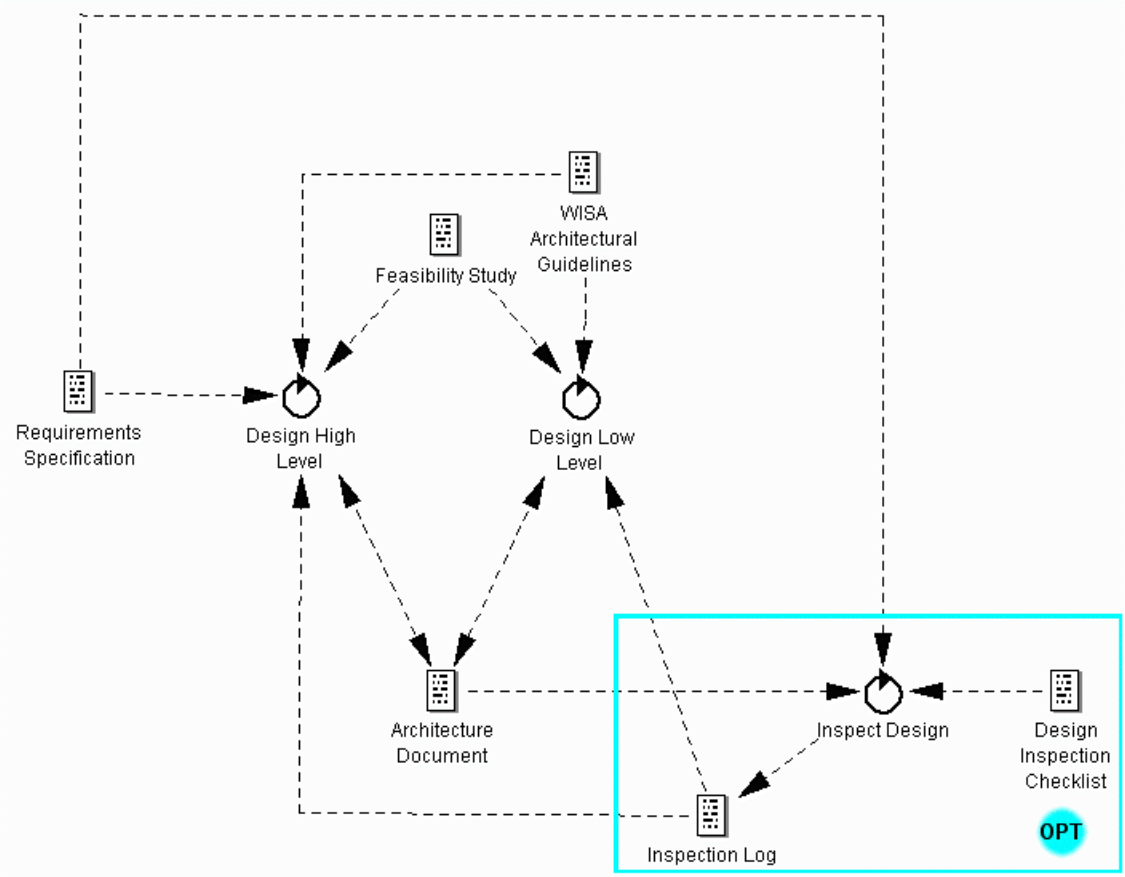

Figure 2-12. Product flow design phase.

2.4.10 Process View: *Roles*

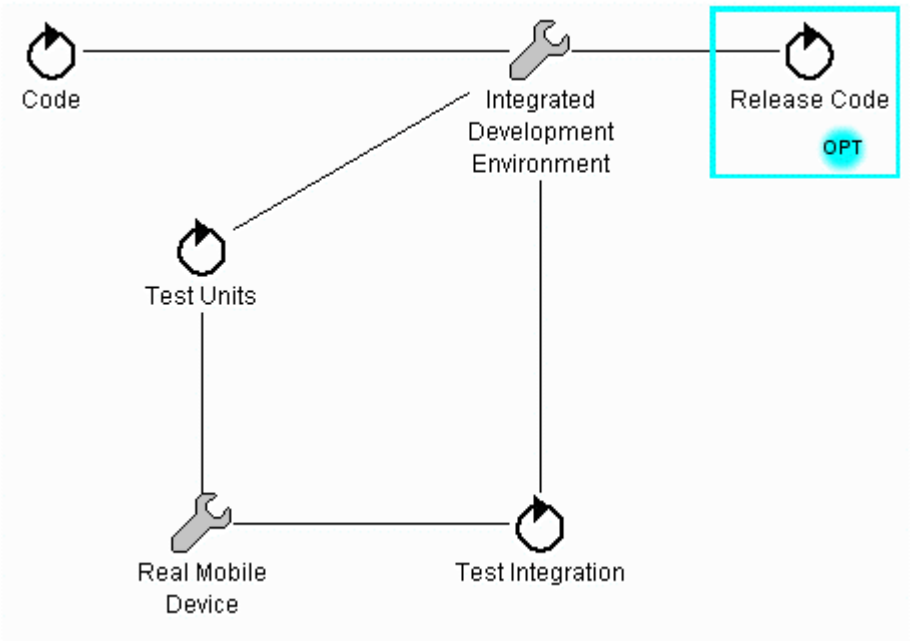

Figure 2-13: Process view. Design phase roles.

2.4.11 Process View: *Tools*

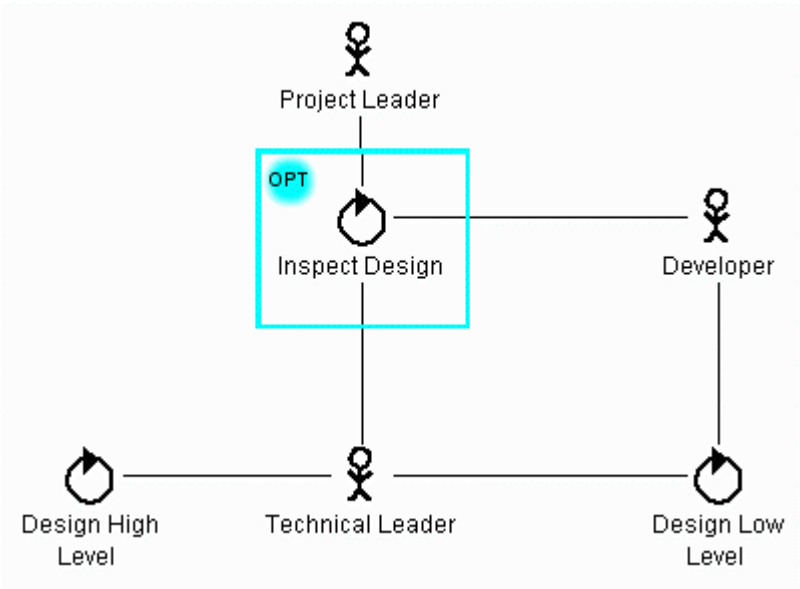

Figure 2-14. Process view: Design phase tools

2.4.12 Involved Roles

The following roles are involved with this activity:

- Developer
- Project Leader
- Technical Leader

2.4.13 Used Tools

This activity uses the following tools:

- Text Editor
- UML Editor

2.4.14 Activity: *Design High Level*

2.4.14.1 Purpose

To identify and model the following:
- Structure and relations of conceptual entities.
- Dynamics and the interactions among supposed services offered by the conceptual entities.
- Anticipated distribution of services in the system execution environment.

2.4.14.2 Type

Common

2.4.14.3 Description

1. Define the conceptual architecture using the WISA (Wireless Internet Service Architecture) Reference Guidelines [22].

2.4.14.4 Input Criteria

1. The requirements document approved by the customer.
2. The results of the feasibility study.
3. A defect from a previous version was identified during the design review and documented in the defect correction plan. The defect was traced back to the design artifact. The conceptual architecture must be reworked.
4. A defect was identified and documented in the inspection log template. The design must be reworked.

2.4.14.5 Exit Criteria

1. A complete design of the conceptual level of the architecture.

2.4.14.6 Product Flow

This activity consumes the following artifacts:

- Requirements Specification
- Feasibility Study
- WISA Architectural Guidelines
- Inspection Log

This activity modifies the following artifacts:

- Architecture Document

This activity does not produce any artifacts.

2.4.14.7 Subactivities

The following are the subactivities of this activity:

- Specify Conceptual Design
- Aquire COTS

2.4.14.8 Product Flow Refinement

The following graph depicts the product flow for this activity:

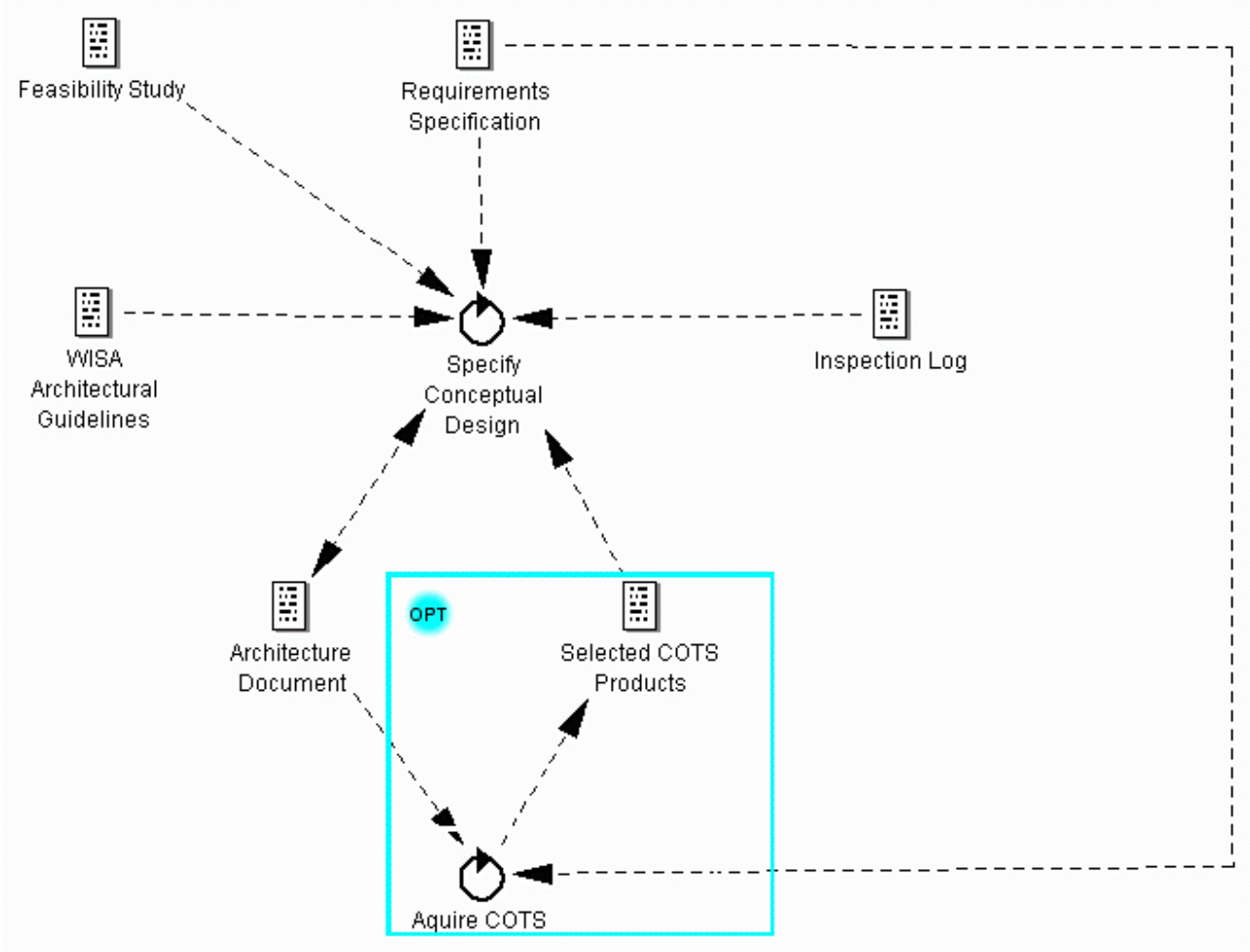

Figure 2-15. Product flow high-level design phase.

2.4.14.9 Involved Roles

The following roles are involved in this activity:

- Technical Leader

2.4.14.10 Used Tools

This activity uses the following tools:

- UML Editor
- Text Editor

2.4.14.11 Activity: *Specify Conceptual Design*

2.4.14.11.1 Purpose

To divide the requirements into logical units.

2.4.14.11.2 Type

Common

2.4.14.11.3 Description

1. Identify new end-user services that belong to the end-user domain category of wireless services depicted in WISA/RA. The category gives an initial list of characteristics, functional and quality requirements.
2. Order quality requirements: the qualities important in achieving the real quality of the end-user service are on top of that priority list.
3. Select the styles and patterns to end-user services that exploit WISA/RA and its basic services according to the identified quality attributes.
4. Use the conceptual structure from WISA/RA to specify the conceptual structure of the new service.

2.4.14.11.4 Input Criteria

1. The requirements document approved by the customer.
2. The results of the feasibility study.
3. A defect was identified and documented in the inspection log template. The design must be reworked.
4. A defect from a previous version was identified and documented in the defect correction plan. The defect was traced back to the design artifact. The design must be reworked.

2.4.14.11.5 Exit Criteria

1. A complete design of the product conceptual structure.

2.4.14.11.6 Product Flow

This activity consumes the following artifacts:

- Requirements Specification
- Feasibility Study
- WISA Architectural Guidelines
- Selected COTS Products
- Inspection Log

This activity modifies the following artifacts:

- Architecture Document

This activity does not produce any artifacts.

2.4.14.11.7 Involved Roles

- Developer

2.4.14.11.8 Used Tools

- UML Editor
- Text Editor

2.4.14.12 **Activity**: ***Acquire COTS***

2.4.14.12.1 Purpose

To identify, evaluate and select the COTS products that meet the architecture.

2.4.14.12.2 Type

Optional

2.4.14.12.3 Description

1. Identify the COTS products.
2. Evaluate the COTS products.
3. Select the COTS products.

2.4.14.12.4 Risk Factors

Organizational risks
- Lack of experience acquiring COTS products.

Risk mitigation hints

Hint 1
Understand and follow PORE [23]. PORE is an elicitation of features of existing COTS software and requirements engineering are conducted in parallel. Eventually a COTS software is selected that almost exactly fits the requirements.

Hint 2
Understand and follow OTSO [24]. Starting from a set of requirements specifying the system component, a decision taxonomy using AHP[25] and a set of measures is defined to select the most suitable COTS component in a given requirements context. The phases are screening on the full set of measures, ranking, detailed evaluation, cost and value estimation, and then the buy decision for a specific COTS software

Hint 3
Understand and follow CAP [25]. Process made up of three parts: Initialization, Execution and Reuse. The first part deals with the acquisition process planning and its cost estimation. Second part provides guidance for performing the COTS assessment (based on the AHP) and taking the make-or-buy decision. The third part is responsible for storing all the information gathered by the other parts in order to decrease the cost of future COTS acquisition processes.

Hint 4
Understand and follow IusWare [26]. The methodology is based is based on the multicriteria decision aid approach and consists of two main phases: design of an evaluation model, application of the model. The design phases can be broken into: identification of relevant actor, identification of evaluation type, definition of a hierarchy of attributes, definition of the measures, choice of an aggregation technique.

Hint 5
Understand and follow Scenario Based COTS Selection [27]. An impact analysis of COTS is carried on considering system scenarios, they are modified under the hypothesis of using different COTS candidates, a new scenario set is produced together with a list of issued encountered during COTS adaptation.

Hint 6
Understand and follow RCPEP [28]. Requirements-driven COTS product evaluation process. This process consists of two phases: trade study, aimed at screening initial candidate products, and hand-on evaluation, which consists of an in-depth evaluation resulting in one (or more) recommended products.

Hint 7
Understand and follow PECA [29]. Is a process for evaluating COTS products made of four phases: planning the evaluation, establishing the criteria, collecting the data, and analyzing the data.

2.4.14.12.5 Input Criteria

1. The requirements specification approved by the customer.
2. A draft version of the product logical structure.

2.4.14.12.6 Exit Criteria

1. A list of selected COTS products.

2.4.14.12.7 Product Flow

This activity consumes the following artifacts:
- Requirements Specification
- Architecture Document

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Selected COTS Products

2.4.14.12.8 Subactivities

The following are the subactivities of this activity:

- Identify COTS Products
- Evaluate COTS Products
- Select COTS Products

2.4.14.12.9 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity

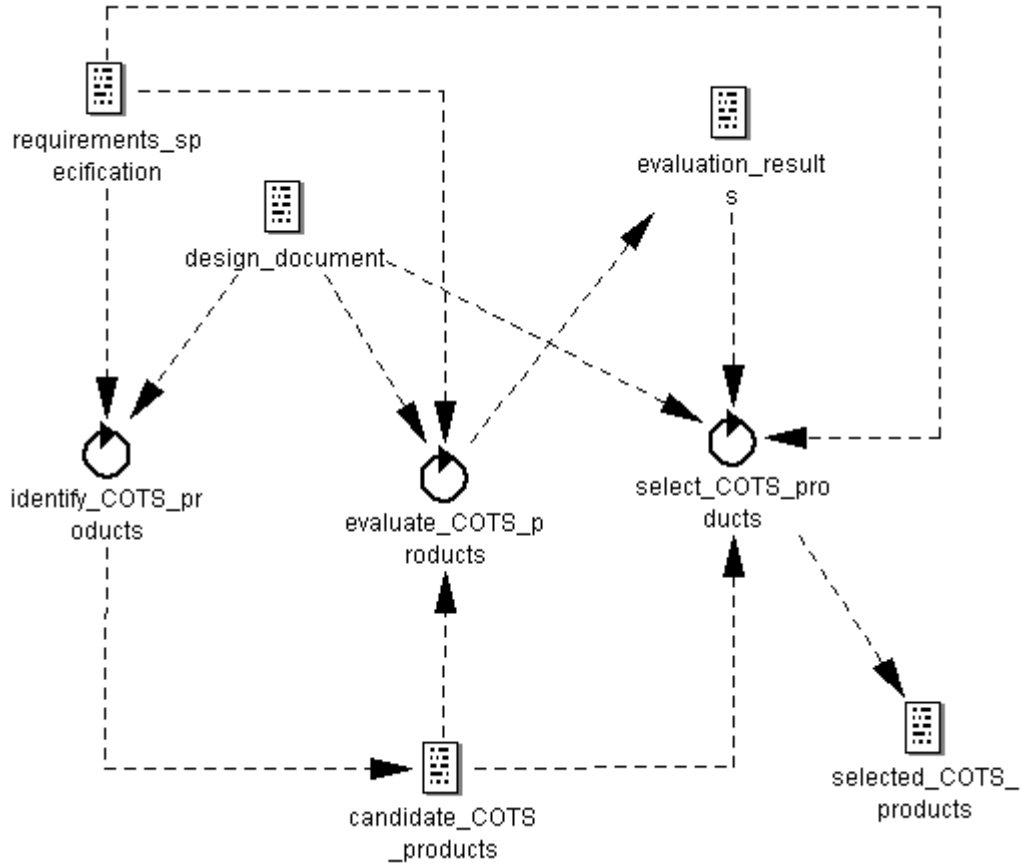

Figure 2-16. Product flow COTS aquisition.

2.4.14.12.10 Involved Roles

- Developer

2.4.14.12.11 Used Tools

- Integrated Development Environment

2.4.14.12.12 Activity: *Identify COTS Products*

2.4.14.12.12.1 Purpose

To find existing components in the market that meet the architecture, enhance the value of the application, accelerate the implementation, and are accessible.

2.4.14.12.12.2 Type

Common

2.4.14.12.12.3 Description

1. Collect a list of COTS products using the knowledge in the organization, mainly about products used in previous projects. If the organization knowledge is not enough, then there is a search for products in the market.
2. Usually the next thing to do is to browse the WEB or ask consultant organizations for reports and evaluations of COTS products. One source are the defined products in WISA.

2.4.14.12.12.4 Input Criteria

1. The requirements specification approved by the customer.
2. A complete design of the conceptual level of the architecture

2.4.14.12.12.5 Exit Criteria

1. A list of previously identified COTS products (called candidate COTS products).

2.4.14.12.12.6 Product Flow

This activity consumes the following artifacts:
- Requirements Specification
- Architecture Document

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Candidate COTS Products

2.4.14.12.13 Activity: *Evaluate COTS Products*

2.4.14.12.13.1 Purpose

To measure a defined set of attributes from each of the possible COTS products that meets the architecture.

2.4.14.12.13.2 Type

Common

2.4.14.12.13.3 Description

1. Define the set of attributes of the products that must be measured. This is done using as a basis the design and the requirements of the application.
2. Take all the candidate products one by one and measure the selected attributes. The measurement may require conducting experiments and setting up test-beds.
3. Produce an evaluation result document that contains the measures of all the attributes for all the products. (a basic Excel table).

For example:

- Product ID 1

Performance (attribute measure 1) = 5 out of 10
Portability (attribute measure 1) = 6 out of 10
Support (attribute measure 1) = 5 out of 10

- Product ID 2

Performance (attribute measure 1) = 9 out of 10
Portability (attribute measure 1) = 8 out of 10
Support (attribute measure 1) = 7 out of 10

2.4.14.12.13.4 Input Criteria

1. The requirements specification approved by the customer.
2. A complete design of the product logical structure.
3. A list of previously identified COTS products (called candidate COTS products).

2.4.14.12.13.5 Exit Criteria

1. A finished evaluation of the candidate COTS products.

2.4.14.12.13.6 Product Flow

This activity consumes the following artifacts:
- Requirements Specification
- Architecture Document
- Candidate COTS Products

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Evaluation Results

2.4.14.12.13.7 Involved Roles

The following roles are involved with this activity:
- Developer

2.4.14.12.13.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Text Editor

2.4.14.12.14 Activity: *Select COTS Products*

2.4.14.12.14.1 Purpose

To rank the list of possible COTS products that meet the architecture.

2.4.14.12.14.2 Type

Common

2.4.14.12.14.3 Description

1. Define the criteria to select the products you need. Here, weights for the attributes are defined. There are several methods to define the higher weights; one is for example to give higher weight values to the most important attributes and the lower weight values to the less important ones.
2. Multiply the results of the attribute measurements by their corresponding defined weights, add the results of the products per COTS product, and select the ones most suitable for your criteria.
An example of the resulting table is shown:
For example:
Weights: Performance: 2, Portability: 7, Support: 10

- Product ID 1

Performance = 5 (attribute measure 1) * 2 = 10
Portability = 6 (attribute measure 1) * 7 = 42
Support = 5 (attribute measure 1) * 10 = 50

Total Product 1: 102

- Product ID 2

Performance = 9 (attribute measure 1) * 2 = 18
Portability = 8 (attribute measure 1) * 7 = 56
Support = 7 (attribute measure 1) * 10 = 70

Total Product 2: 144

2.4.14.12.14.4 Input Criteria

1. The requirements specification approved by the customer.
2. A complete design of the product logical structure.
3. A list of previously identified COTS products (called candidate COTS products).
4. A finished evaluation of the candidate COTS products.

2.4.14.12.14.5 Exit Criteria

1. A list of selected COTS products.

2.4.14.12.14.6 Product Flow

This activity consumes the following artifacts:
- Requirements Specification
- Architecture Document
- Candidate COTS Products
- Evaluation Results

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Selected COTS Products

2.4.14.12.14.7 Involved Roles

The following roles are involved with this activity:
- Developer

2.4.14.12.14.8 Used Tools

This activity uses the following tools:
- Text Editor

2.4.15 Activity: *Design Low Level*

2.4.15.1 Purpose

To idenfy and model the following:
- Concrete structural elements and relationships among them.
- The dynamics of a system and the interactions among classes and/or among components.
- The actual execution environment (network and business structures) in which a system will be operated.
-The interfaces between the concrete components.

2.4.15.2 Type

Common

2.4.15.3 Description

1. Define the concrete architecture using the WISA (Wireless Internet Service Architecture) Reference Guidelines [22].

2.4.15.4 Input Criteria

1. A complete design of the conceptual architecture.
2. Guidelines for designing the concrete architecture.
3. A defect was identified during the design review and was documented in the inspection log template. The concrete architecture must be reworked.
4. A defect from a previous version was identified and documented in the defect correction plan. The defect was traced back to the design artifact. The design must be reworked.

2.4.15.5 Exit Criteria

1. A complete design of the conceptual and concrete levels of the architecture.

2.4.15.6 Product Flow

This activity consumes the following artifacts:
- Feasibility Study
- WISA Architectural Guidelines
- Inspection Log

This activity modifies the following artifacts:
- Architecture Document

This activity does not produce any artifacts.

2.4.15.7 Involved Roles

The following roles are involved in this activity:
- Developer
- Technical Leader

2.4.15.8 Used Tools

This activity uses the following tools:
- UML Editor
- Text Editor

2.4.16 Activity: *Inspect Design*

2.4.16.1 Purpose

To uncover design problems noticeable in the design document.

2.4.16.2 Type

Optional

2.4.16.3 Description

The steps are taken from the script INS of the book TSPi [30].
1. A moderator and reviewers are selected for the inspection.
2. The moderator makes sure that the product is ready for inspection.

3. The moderator describes the inspection process.
The creator presents briefly the product to the reviewers.
The reviewers select criteria for the inspection.
Criteria can be: Operation, recovery, maintenance, security, installation, size, performance, and others (It would be interesting to see criteria that apply to Wireless Internet Services development as well).
4. The moderator sets date and time of the inspection.
5. The reviewers separately make detailed product reviews.
6. The reviewers mark defects found on the data collection sheets.
7. They record their preparation time for the inspection meeting.
8. The moderator opens the inspection meeting. If any reviewer is not prepared, he reschedules the meeting.
9. The moderator walks through the design.
9.1 Has the reviewers describe every problem found, and registers the problem data.
10. The inspection team decides whether a re-inspection is warranted, who should do it, when, and how to verify the defect corrections
11. The creator:
- Makes repairs and updates the product documentation
- Calls for a new inspection.

2.4.16.4 Input Criteria

1. A complete design of the logical structure, its components, and its interfaces.
2. A completed requirements specification.
3. Design review checklist.

2.4.16.5 Exit Criteria

1. The completed inspection log with the results of the inspection activity.

2.4.16.6 Product Flow

This activity consumes the following artifacts:
- Requirements Specification
- Architecture Document
- Design Inspection Checklist

This activity produces the following artifacts:
- Inspection Log

2.4.16.7 Involved Roles

The following roles are involved with this activity:
- Developer
- Project Leader
- Technical Leader

2.4.16.8 Used Tools

This activity uses the following tools:
- Text Editor
- UML Editor

2.5 *WISEP / CODING PHASE*

2.5.1 Purpose

To produce and integrate the code that implements the architecture design document and meets the requirement specification.

2.5.2 Type

Common

2.5.3 Description

1. The code is produced and tested.
2. The code is integrated.
3. (Optional) The code is released.

2.5.4 Risk Factors

Technological risks
- Lack of experience with available programming languages
- Small screens
- Limited keyboards
- Diversity of input mechanisms
- Limited device power capability
- Limited non-volatile memory storage

Risk mitigation hints
Hint 1
Internet services for providing financial, weather, or sport information to clients, i.e., services that require little user interaction "thin clients" can in theory be deployed in wireless Devices using the Wireless Application Protocol (WAP). At the moment the WAP 1.x and 2.0 standards are available [31]. WAP 1.x uses the Wireless Markup Language (WML) for document formating. WML is a language similar to HTML, specially designed for small clients with small screens and low bandwidth. A web site developed with HTML does not need a complete architecture rework of the service in order to be translated into WML, but maintenance can be a heavy duty because every modification to the desktop version should also be made in the mobile version.
Hint 2
WAP 2.0 uses the extensible hypertext markup language (XHTML) for formatting the document. In theory, WAP2.0 allows developers to create richer applications that handle multimedia and animation, among other features. XHTML can be displayed by almost all available browers, but not all HTML features can be converted into XHTML. Be aware of the performance of XHTML.
Hint 3
Security for "thin clients" can be assured through the Wireless Transport Layer Security (WTLS).
Hint 4
cHTML is the content development language for i-mode (NTT Docomo's Wireless Service). cHTML is also similar to HTML, but is optimized for wireless networks and devices.
Hint 5
In the case of J2ME applications "fat clients", power and memory constrains make security a challenge, because the libraries consume much from the device resources. Be aware of new J2ME features and how do they handle these constrains.
Hint 6
Even though, memory storage of mobile devices, is a constraint that tends to improve with technology evolution, techniques for reducing the size of compiled code to be stored in the device memory are welcome. Obfuscation, for example, is a technique to protect software and optimize its execution [32].
Hint 7
Today some wireless devices can be programmed using some sort of C-like language, but C is not a cross-platform language and therefore, portability among different hardware architectures is lost. A dedicated fat client should be deployed for every possible platform, slowing down time-to-market of the service and increasing costs.
Organizational risks
- Lack of experts for programming wireless Internet services.
- Lack of experience testing code on mobile devices.

Hint 1
Pair programming is an extreme programming technique where two developers produce code in one machine [33]. One person concentrates on the strategy to produce the code and the other on whether the approach could work, how it could be simplified, and has the control of the computer
Hint 2
Most of the mobile devices vendors provide emulator environments that can be used to test the functionality of applications (e.g., Sun, Motorola, Siemens, Nokia, RIM). That is the case of Sun's J2MEs Wireless Toolkit, which is a set of tools that provides developers with the emulator environment needed to implement applications targeted at CLDC/MIDP compliant mobile phones and entry level. It is important to notice though, that emulators are not 100% reliable. Due to the fact that emulators are not 100% reliable, it would be better to make specific tests on cell phones, such as network modules. They must been immediately tested on cell phones since they are critical for the rest of the application.

2.5.5 Input Criteria

1. The requirements document approved by the customer.
2. A complete design of the conceptual and concrete levels of the architecture.
3. A defect from a previous version was identified and documented in the defect correction plan. The defect was traced back to the code artifact. The code must be reworked.

2.5.6 Exit Criteria

1. An integrated product.

2.5.7 Product Flow

This activity does not consume any artifacts.
This activity modifies the following artifacts:

- Requirements Specification
- Architecture Document

This activity produces the following artifacts:

- Source Code

2.5.8 Subactivities

The following are the subactivities of this activity:

- Code
- Test Units
- Test Integration
- Release Code

2.5.9 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity.

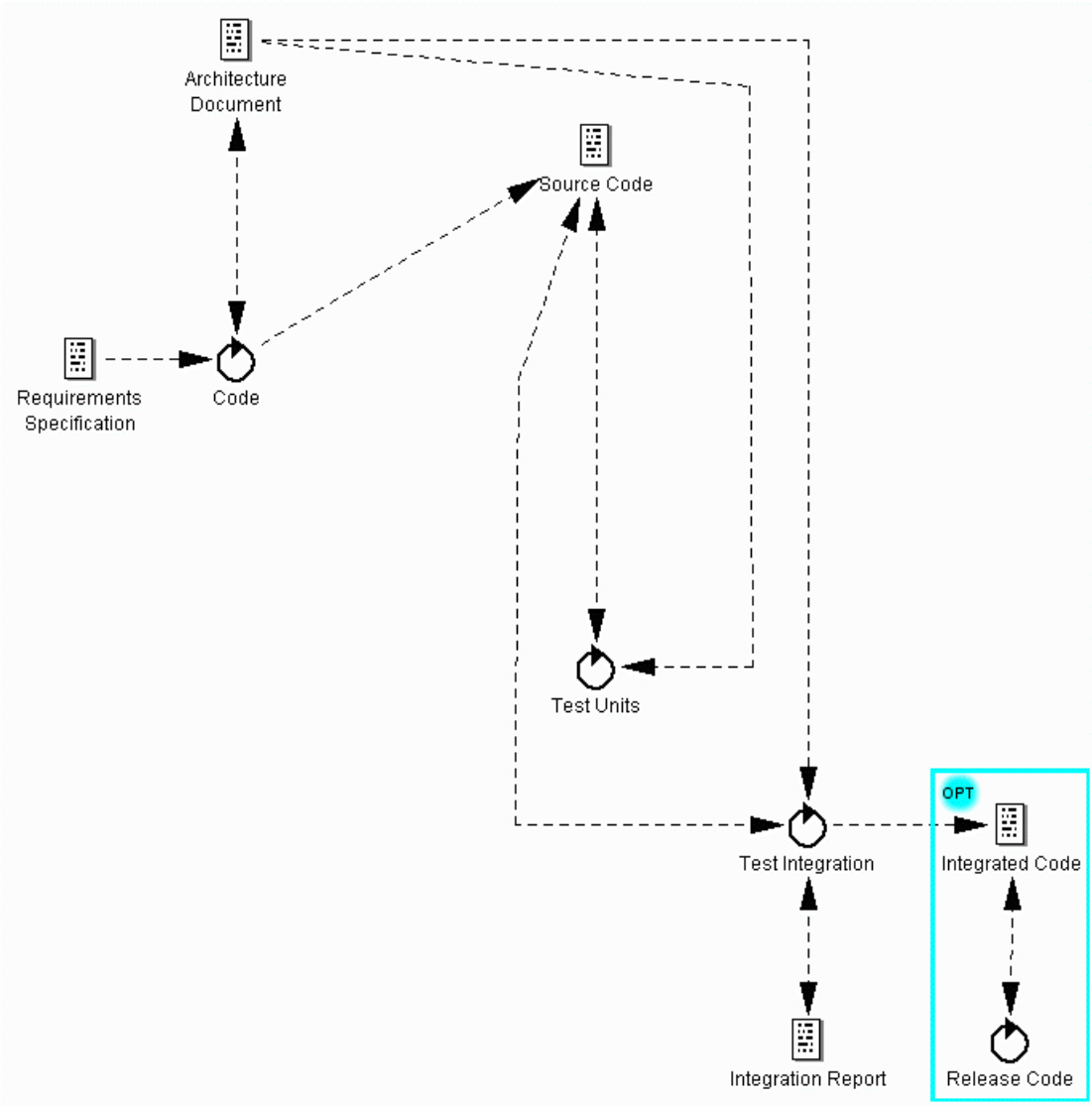

Figure 2-17. Product Flow Coding Phase.

2.5.10 Process View: *Roles*

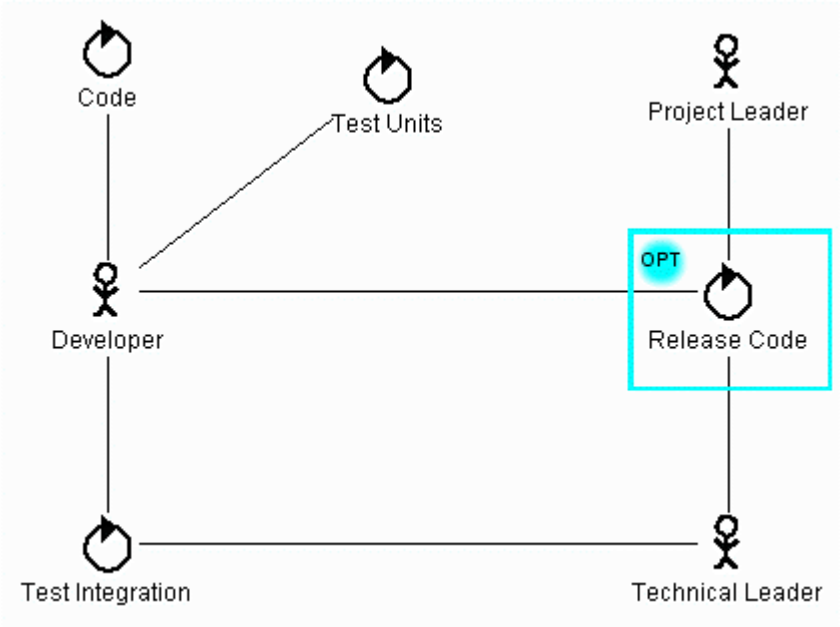

Figure 2-18. Process view: Coding Phase Roles

2.5.11 Process View: *Tools*

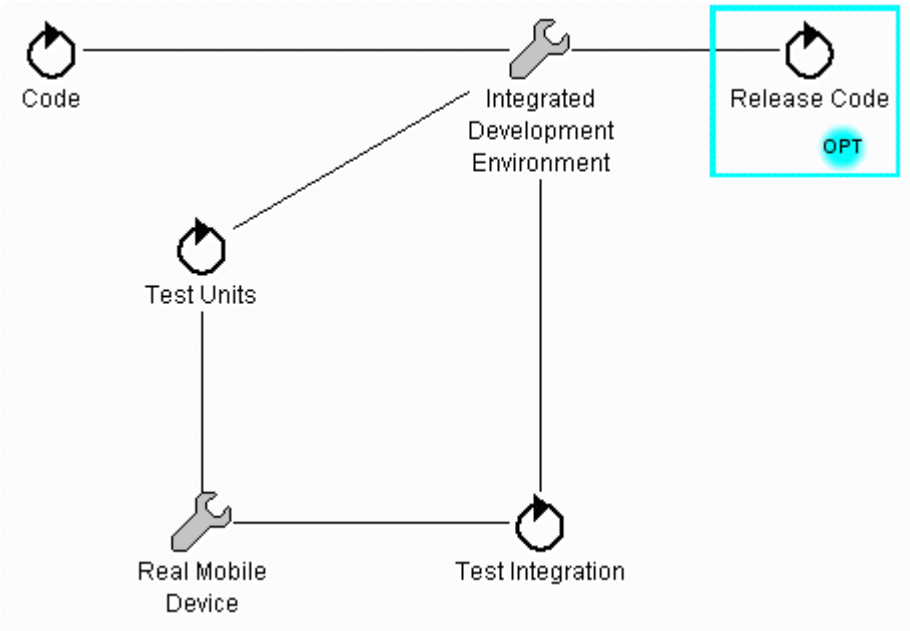

Figure 2-19. Process view: Coding Phase Tools

2.5.12 Involved Roles

The following roles are involved with this activity:

- Developer
- Project Leader

- Technical Leader

2.5.13 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.5.14 Activity: *Code*

2.5.14.1 Purpose

To produce the units of code that implements the architecture design document and meets the requirement specification.

2.5.14.2 Type

Common

2.5.14.3 Description

1. Implement the design in the correspondent units of code.

2.5.14.4 Input Criteria

1. The requirements document approved by the customer.
2. A complete design of the conceptual and concrete levels of the architecture.
3. A defect from a previous version was identified and documented in the defect correction plan. The defect was traced back to the code artifact. The code must be reworked.

2.5.14.5 Exit Criteria

1. Units of code that complies with the coding standards.

2.5.14.6 Product Flow

This activity consumes the following artifacts:
- Requirements Specification

This activity modifies the following artifacts:
- Architecture Document

This activity produces the following artifacts:
- Source Code

2.5.14.7 Involved Roles

The following roles are involved in this activity:
- Developer

2.5.14.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment

2.5.15 Activity: *Test Units*

2.5.15.1 Purpose

To test the units of code until all tests run without errors.

2.5.15.2 Type

Common

2.5.15.3 Description

1. Test the units of code.
2. Fix all defects found: Modifications to the requirements or design document are possible.

2.5.15.4 Input Criteria

1. Units of code that comply with the coding standards.
2. A complete design of the conceptual and concrete levels of the architecture.

2.5.15.5 Exit Criteria

1. Units of code without errors.

2.5.15.6 Product Flow

This activity consumes the following artifacts:

- Architecture Document

This activity modifies the following artifacts:

- Source Code

This activity does not produce any artifacts.

2.5.15.7 Involved Roles

The following roles are involved in this activity:

- Developer

2.5.15.8 Used Tools

This activity uses the following tools:

- Integrated Development Environment
- Real Mobile Device

2.5.16 Activity: *Test Integration*

2.5.16.1 Purpose

To integrate the units of code that implements the design and meets the requirements specification.

2.5.16.2 Type

Common

2.5.16.3 Description

1. Verify that all needed parts are on hand.
2. Build the product
3. When defects are found it should be determined whether integration should continue.
4. Every defect found is recorded in the integration report and reviewed to determine:

- Where similar defects may remain in the product.
- How and when to find and fix these defects.

2.5.16.4 Input Criteria

1. Units of code (modules, functions, methods) without errors.
2. Feasibility study results.
3. A complete design of the logical structure, the components and its interfaces.
4. A defect from a previous version was identified and documented. The defect was traced back to the Integated Code artifact. The Integrated Code must be reworked.

2.5.16.5 Exit Criteria

1. An integrated and tested product.

2.5.16.6 Product Flow

This activity consumes the following artifacts:
- Architecture Document

This activity modifies the following artifacts:
- Source Code
- Integration Report

This activity produces the following artifacts:
- Integrated Code

2.5.16.7 Involved Roles

The following roles are involved in this activity:
- Developer
- Technical Leader

2.5.16.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.5.17 Activity: *Release Code*

2.5.17.1 Purpose

To establish a product baseline and enter it in a configuration management system.

2.5.17.2 Type

Optional

2.5.17.3 Description

1. A baseline number is associated to the integrated code.
2. The objectives and features of the code are released.
3. The code is entered in the configuration management system

2.5.17.4 Input Criteria

1. An integrated and tested product.

2.5.17.5 Exit Criteria

1. A baselined product.

2.5.17.6 Product Flow

This activity does not consume any artifacts.
This activity modifies the following artifacts:

- Integrated Code

This activity does not produce any artifacts.

2.5.17.7 Involved Roles

The following roles are involved in this activity:

- Developer
- Project Leader
- Technical Leader

2.5.17.8 Used Tools

This activity uses the following tools:

- Integrated Development Environment

2.6 WISEP / PLAN TESTING PHASE

2.6.1 Purpose

To produce test cases for the each of the requirements specified and settle the needed hardware and software in order to execute the test cases.

2.6.2 Type

Optional

2.6.3 Description

1. Plan the tests to check the structure, functionality, performance, content, and usability of the service.
2. Set the needed hardware and software in order to run the planned tests.

2.6.4 Risk Factors

Technological risks
- Diversity of wireless network standards
- Lack of experience with new charging and billing models
- Small screens
- Limited keyboards
- Diversity of input mechanisms
- Limited device power capability
- Limited non-volatile memory storage

Risk mitigation hints
Hint 1
Central-Control Wireless Emulators abstract the entire mobile wireless network to a model with a set of parameters thus emulating end-to-end applications and protocols. These emulators apply network conditions and traffic dynamics to each packet that is passing by to reproduce the network effects thus testing the performance of the applications and protocols. This type of emulation is conducted by connecting mobile hosts such as handheld devices, computers to the central-control emulator. Some general-purpose network emulators falling into this category are ONE [34], Dummynet [35].
Hint 2
VINT/NS [36] is one simulation combined wireless emulators. The emulation facility in VINT/NS is able to capture and direct traffic into the simulator. Within the simulator, protocol modules, algorithms, and visualization tools can be incorporated into the emulator in an automatic fashion. In addition, arbitrary

mobility can be generated with the help of the simulator. The advantage of this approach is that it offers a large amount of simulation resources in the central simulator, compared with the central-control approach that only has a limited number of network parameters available.
Hint 3
Trace based mobile network emulator [37] also emulates the characteristics such as performance, bandwidth in a real environment. However the approach in this emulation is different. This approach consists of three distinct phases: data collection phase, trace distillation phase, and modulation phase.
Hint 4
Another emulator named flying emulator [38] to build and test application level software for wireless mobile computing, emulates the physical mobility of wireless devices by using the logical mobility of software-based emulators of the devices and target software. The emulator is implemented as a mobile agent; it carries dynamically the target software to each of the sub-networks to which its device is connected on behalf of the device, permitting the software to interact with other servers in the current sub-network. That is, it can test software designed to run on a wireless device in the same way as if the software were disconnected from the network, moved with the device, and reconnected to and operated on another network.
Hint 5
One distributed network emulator system, EMPOWER [39], provides a mechanism to emulate the mobility of a wireless network in a wire line network. The preliminary results of emulating node mobility of wireless networks using EMPOWER are encouraging. EMPOWER allows the user to define packet latency and bandwidth as parameters and test a given topology wireless network.
Hint 6
A description of the Model Based Testing technique, and its application for testing a Pocket PC application can be found in [40]. The technique uses finite state machines and directed graphs or state transition diagrams as a basis for testing the functionality of the application. Benefits of model-based testing are the possibility to automate, and the fact that the structure of states and transitions is written, which gives a general understanding to all team members on how the application should work. It is still a topic of research to find out if model-based testing is suitable for finding faults, due to the fact that the effort invested by developers on building the model is not depreciable.
Hint 7
Complete network simulation including the radio frequency [41] includes the complete simulation of base station system and core network elements to establish, maintain and delete the connection between the device and the server. The client application running on a handheld containing the wireless modem, talks to the Network Simulator Box on the Radio frequency (RF) interface and simulator box talks to the application server on the IP interface. In this way end-to-end push or pull application can be triggered with the simulator acting as the mobile network bearer.
Hint 8
Band network behaviour simulation on IP level [42] captures the IP packets between the client and the server and introduces the network related interruptions or error scenarios.
Hint 9
Markov models are mathematical models, which can be used for the validation of the usability of the mobile devices [43]. This mathematical construct is helpful in analyzing the usability of push button devices such as mobile phones, PDA's and etc. This model is based on the finite state machines. Finite state machines represent the whole system as a set of states and transitions and as well-defined mathematical objects, it is possible to perform complex reasoning on the model to produce reproducible, quantitative results.

2.6.5 Input Criteria

1. The requirements document approved by the customer.

2.6.6 Exit Criteria

1. A completed set of test cases for checking the functionality, usability, and performance of the system.
2. The required hardware and software to test the product.

2.6.7 Product Flow

This activity consumes the following artifacts:

- Requirements Specification

This activity does not modify any artifacts.
This activity produces the following artifacts:

- Test Plan
- Test Framework

2.6.8 Subactivities

The following are the subactivities of this activity:

- Plan Tests
- Build Test Framework

2.6.9 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity. You can click on them to get a full size view:

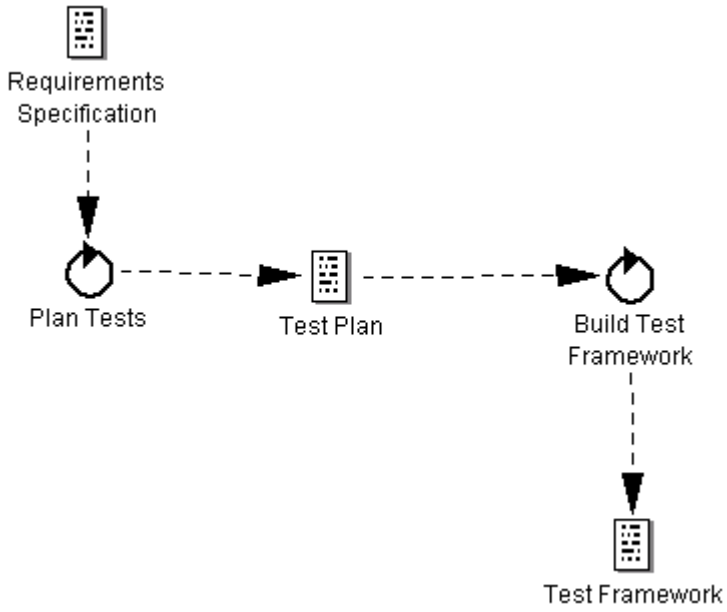

Figure 2-20. Product Flow: Plan Testing Phase

2.6.10 Process View: *Roles*

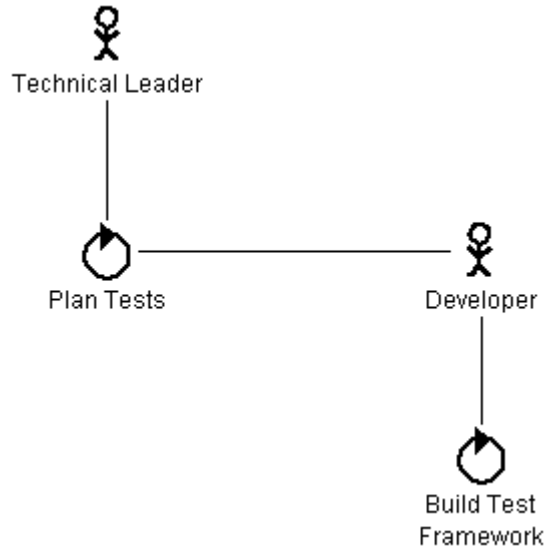

Figure 2-21. Process view: Plan Testing Phase Roles.

2.6.11 Process View: *Tools*

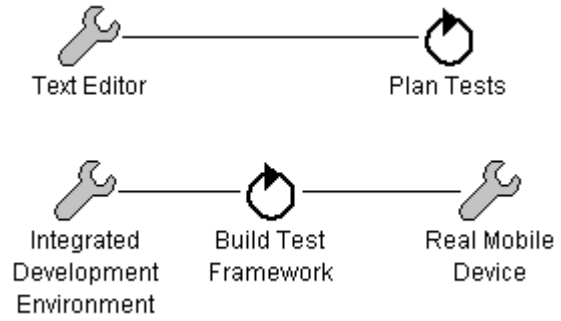

Figure 2-22. Process view: Plan Testing Phase Tools.

2.6.12 Involved Roles

The following roles are involved with this activity:

- Developer
- Technical Leader

2.6.13 Used Tools

This activity uses the following tools:

- Integrated Development Environment
- Real Mobile Device
- Text Editor

2.6.14 Activity: *Plan Tests*

2.6.14.1 Purpose

To produce a test case or a set of test cases for the each of the requirements specified.

2.6.14.2 Type

Common

2.6.14.3 Description

1. Design test cases to check the functionality, performance, content, and usability of the service.
FUNCTIONAL TESTING
- Define test cases to check that the service offered is delivered by the application accordingly.
PERFORMANCE TESTING
- Define test cases to check the performance of the service for the load levels designed.
-Define test cases to check the performance of the service for overload levels.
USABILITY TESTING
-Define test cases to check UI usability.
2. Review the test cases in order to avoid duplication and inconsistencies.

2.6.14.4 Input Criteria

1. The requirements document approved by the customer.

2.6.14.5 Exit Criteria

1. A complete test plan with the sequence of test cases and a description of the needed environment to test the product (Hardware and software).

2.6.14.6 Involved Roles

The following roles are involved with this activity:
- Developer
- Technical Leader

2.6.14.7 Used Tools

This activity uses the following tools:
- Text Editor

2.6.15 Activity: *Build Test Framework*

2.6.15.1 Purpose

To set up the testing framework needed for executing the test plans.

2.6.15.2 Type

Common

2.6.15.3 Description

1. Identify the hardware and software to be used.
2. Set up the physical environment. (Connect server with clients through the needed network)
3. Identify the use or creation of emulators.
4. Code the missing parts.

5. Integrate the testing framework and perform tests to check for defects or problems.

2.6.15.4 Input Criteria

1. A complete test plan with the sequence of test cases and a description of the environment needed to test the product (hardware and software).

2.6.15.5 Exit Criteria

1. The hardware and software required to test the product is settled.

2.6.15.6 Product Flow

This activity consumes the following artifacts:
- Test Plan

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Test Framework

2.6.15.7 Involved Roles

The following roles are involved in this activity:
- Developer

2.6.15.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.7 WISEP / TESTING PHASE

2.7.1 Purpose

To execute the designed test cases that validate that the specified requirements meet are met by the service implementation.

2.7.2 Type

Common

2.7.3 Description

1. The system test is executed.
2. The system is tested by the end users.
3. The results from end users testing are analyzed and a test report is produced.

2.7.4 Risk Factors

Technological risks
- Diversity of wireless network standards
- Lack of experience with new charging and billing models
- Small screens
- Limited keyboards
- Diversity of input mechanisms
- Limited device power capability
- Limited non-volatile memory storage

Risk mitigation hints

Hint 1
Visualization techniques can also be used to analyze the past behaviour of a site, and to understand the impact of new changes. In order to improve the usability of a web site, and to locate problem areas, The visualization-based approach [44], underlines the importance of using visualization techniques to understand the behavior of users on a web site and to identify unreachable places.
Hint 2
The Card Sorts technique is proposed by [45] for eliciting quality measures of web pages. It is a technique based on a personal construct theory, whose objective is to elicit and ensure the validity of a measure for a fuzzy attribute like quality in a new field such as the Internet. It provides a systematic way to elicit quality measures that the stakeholders consider important. In a new domain like wireless Internet, this can be of great help, because it minimizes the suppositions about the stakeholder's usability preferences.
Hint 3
Usability testing is an iterative process that involves testing the site and then using the test results to change the site to better meet users' needs. The best process is to try out a prototype with a few users, fix it, and test it again.
Hint 4
A typical approach for testing usability, is that users one at a time or two working together use the service to perform tasks, while one or more people watch, listen, and take notes.
well on the site (or other product or service). This approach proposes to make the following questions
Do users complete a task successfully?
If so, how fast do they do each task?
Is that fast enough to satisfy them?
What paths do they take in trying?
Do those paths seem efficient enough to them?
Where do they stumble?
What problems do they have?
Where do they get confused?
What words or paths are they looking for,that are not now on the site? Please take a look at the complete set of guidelines at: (http://usability.gov/methods/usability_testing.html)
Hint 5
Concerning how to test the usability of wireless Internet services, the web usability assessment model [46] includes eleven usability attributes, which have been identified as significant in assessing a customer's perceived usability. The usability attributes include design layout, navigation, personalization, design consistency, design standards, reliability, security, performance, information content, accessibility, and customer service. An automated usability-testing tool named Usability Enforcer tool based on the web usability assessment model implements a set of usability rules for a targeted customer profile, specified computing environment and the strategic goals of the wireless application.
Hint 6
The software that runs on mobile devices can also be validated with the amount of power consumed by the components of the device when the application is in use [47]. Applications interact not only with the display, but also with various other hardware devices: processor, memory, network interface, and possibly hard drive. All these devices consume energy during operation. This gives an opportunity to monitor and record power levels during the test suite execution. The recorded power levels can then be used to validate energy requirements. Energy consumption requirements depend on a concrete software and hardware system. The efficient energy consumption requirement is that only hardware devices associated with active software functions should be active.

2.7.5 Input Criteria

1. An integrated product.
2. A completed set of test cases for checking the functionality, usability, and performance of the system.
3. The required hardware and software to test the product.

2.7.6 Exit Criteria

1. A complete report including the results from all tests.
2. A released, integrated, and user tested product.
3. A complete plan for fixing the defects in the actual or later version of the product.

2.7.7 Product Flow

This activity consumes the following artifacts:

- Requirements Specification
- Architecture Document
- Source Code
- Test Plan
- Test Framework

This activity modifies the following artifacts:

- Test Report

This activity produces the following artifacts:

- Product Tested

2.7.8 Subactivities

The following are the subactivities of this activity:

- Test System
- Test Acceptance
- Test Usability
- Analyze Defect

2.7.9 Product Flow Refinement

The following graph(s) depict the product flow refinement for this activity. You can click on them to get a full size view:

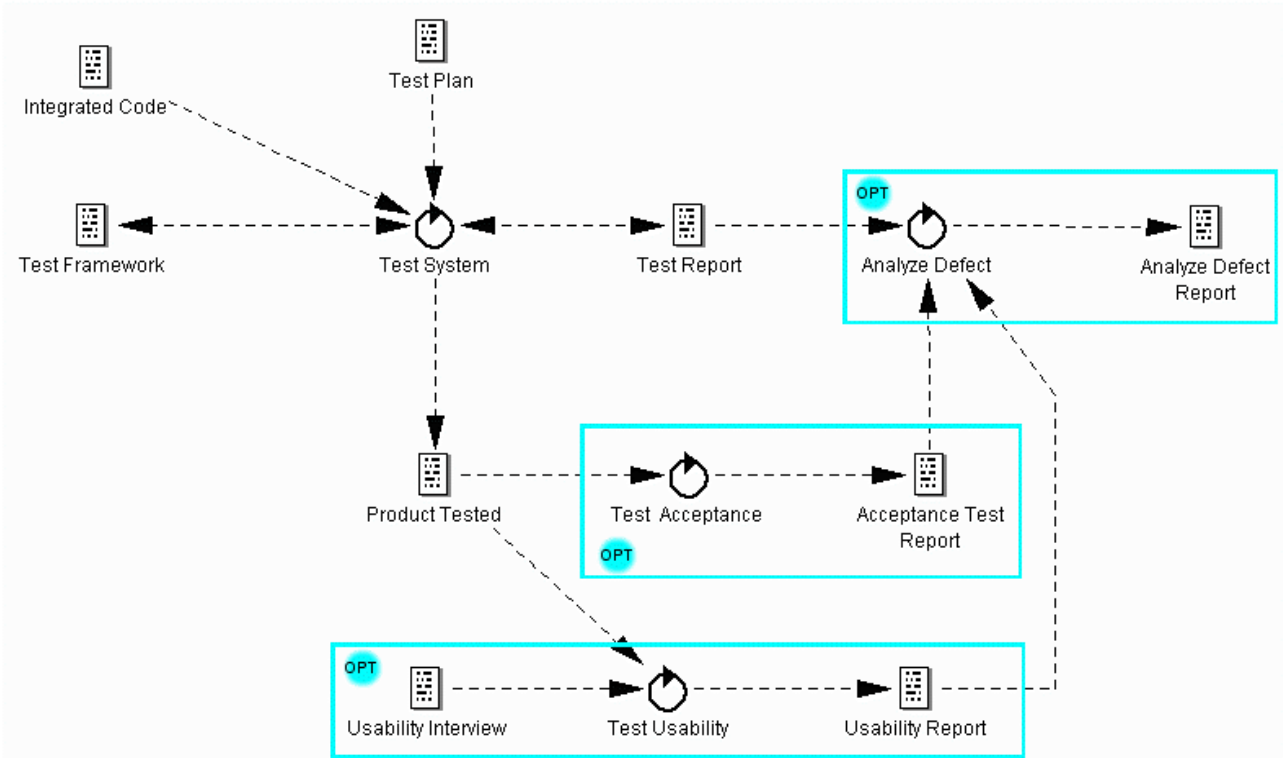

Figure 2-23. Product flow: Testing Phase.

2.7.10 Process View: *Roles*

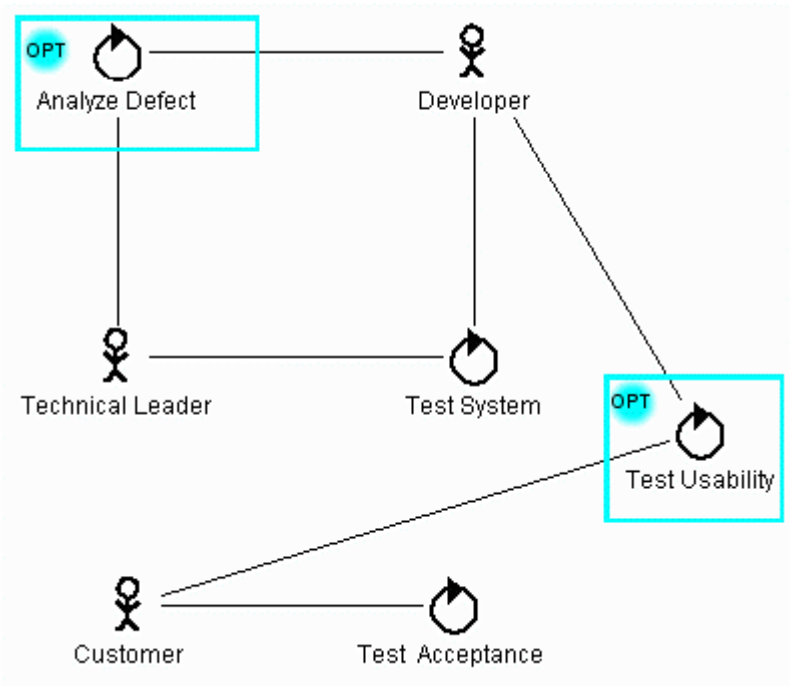

Figure 2-24. Process view: Testing Phase Roles

2.7.11 Process View: *Tools*

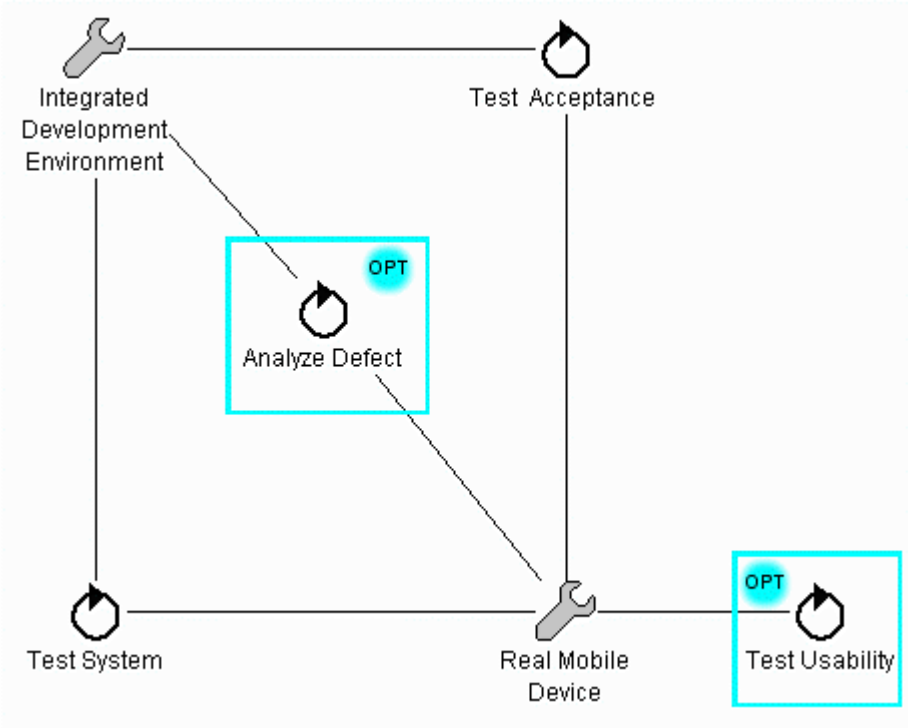

Figure 2-25. Process view: Testing Phase Tools

2.7.12 Activity: *Test System*

2.7.12.1 Purpose

To validate that the implemented system meets the specified requirements.

2.7.12.2 Type

Common

2.7.12.3 Description

1. Test the functionality of the system (integrated on cell phone, laptop, pda, etc..).
2. Test the product for normal and stress conditions.
3. Test the product for installation, conversion and recovery.
4. Execute regression tests on the system.
5. Record all test activities.

2.7.12.4 Input Criteria

1. A released and integrated product.
2. A completed set of test cases for checking the functionality, usability, and performance of the system.
3. The required hardware and software to test the product.

2.7.12.5 Exit Criteria

1. A released and system tested product.
2. Report with record from the performed test plans

2.7.12.6 Product Flow

This activity consumes the following artifacts:
- Integrated Code
- Test Plan

This activity modifies the following artifacts:
- Source Code
- Test Framework
- Test Report

This activity produces the following artifacts:
- Product Tested

2.7.12.7 Involved Roles

The following roles are involved in this activity:
- Developer
- Technical Leader

2.7.12.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.7.13 Activity: *Test Acceptance*

2.7.13.1 Purpose

To facilitate the friendly customers uncover problems or deficiencies of the integrated and tested product.

2.7.13.2 Type

Optional

2.7.13.3 Description

1. Deliver a beta version to a set of friendly customers.
2. Collect the results from tests.
3. Create a formal report with the results from the tests.

2.7.13.4 Input Criteria

1. A released and tested product.

2.7.13.5 Exit Criteria

1. Report with the problems and suggestions found by the friendly customers.

2.7.13.6 Product Flow

This activity consumes the following artifacts:
- Product Tested

This activity does not modify any artifacts.
This activity produces the following artifacts:

- Acceptance Test Report

2.7.13.7 Involved Roles

The following roles are involved in this activity:
- Customer

2.7.13.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.7.14 Activity: *Test Usability*

2.7.14.1 Purpose

To interview the users about possible usability problems.

2.7.14.2 Type

Optional

2.7.14.3 Description

This technique is based in Nielsen's [48] in which experts guided by a set of usability principles known as heuristics evaluate whether user-interface elements such as dialog boxes, menus, navigation structure, online help, etc conform to the principles.
5 evaluators are recommended.
1. The briefing session: Experts are told what to do. A prepared script is useful as a guide and to ensure each person receives the same briefing.
2. The evaluation period: Each expert typically spends 1-2 hours independently inspecting the product using the heuristics for guidance. The experts need to take at least two passes through the interface. The first pass gives a feel of the flow interaction and the product scope, the second allows the evaluator to focus on specific interface elements in the context of the whole products. Preplanned tasks may be helpful.
2.1. While working on the interfaces the evaluator must record the problems.
3. The debriefing session in which experts come together to discuss their findings and to prioritize the problems they found and suggestions.

2.7.14.4 Input Criteria

1. A released and tested product.
2. An interview template

2.7.14.5 Exit Criteria

1. A completed usability report.

2.7.14.6 Product Flow

This activity consumes the following artifacts:
- Product Tested
- Usability Interview

This activity does not modify any artifacts.
This activity produces the following artifacts:
- Usability Report

2.7.14.7 Involved Roles

The following roles are involved in this activity:
- Customer

2.7.14.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.7.15 Activity: *Analyze Defect*

2.7.15.1 Purpose

To decide how to handle and when to handle the problems found during previous testing activities.

2.7.15.2 Type

Optional

2.7.15.3 Description

This activity is performed in parallel with the acceptance tests. If a defect found is considered to be serious, a lists of tasks to solve it is created and the responsible assigned. Replanning concerning the delivery of the product to the market must be made. If the defect is considered to be minor, then a requirement for the new version of the product is generated, therefore, no delays on scheduled delivery are assumed.

2.7.15.4 Input Criteria

1. Report with record from the performed test activities(cases).
2. Report with the problems and suggestions found by the friendly customers.

2.7.15.5 Exit Criteria

1. A complete plan for fixing the defects in the actual or later version of the product.

2.7.15.6 Product Flow

This activity consumes the following artifacts:
- Test Report
- Usability Report

Acceptance Test Report
This activity does not modify any artifacts.
This activity produces the following artifacts:
- Analyze Defect Report

2.7.15.7 Involved Roles

The following roles are involved in this activity:
- Developer
- Technical Leader

2.7.15.8 Used Tools

This activity uses the following tools:
- Integrated Development Environment
- Real Mobile Device

2.8 *WISEP* / *ARTIFACTS*

2.8.1 Artifact: *Requests From Customer*

2.8.1.1 Purpose

To hold the requiests of the customer.

2.8.1.2 Description

The requests from customer can be produced in two different situations: New service or change to a previous service.
New service: In this case the document is usually a description of a new service designed for one specific mobile device. The document is elaborated by the marketing division and describes a wished user interface and the functionality of the service.
Change to a previous service: Bugs, or suggestions for improvement found in previous releases of a service can be documented as customer requirements by customers.

2.8.1.3 Product Flow

This artifact is not produced by any activity.
This artifact is modified by the following activities.

- Analyze UI Feasibility
- Select Feasible Requests

This artifact is used by the following activities:

- Requirements Phase
- Select Requirements
- Write Scenarios
- Write Requests

2.8.2 Artifact: *Requirements Specification*

2.8.2.1 Purpose

To hold the specification of the set of requirements to be implemented in a convenient form.

2.8.2.2 Description

Structure of the document
1. Service architecture: It provides the general framework on which the service is based, and general constraints to be considered.
2. Service functional requirements: describe requirements at the level of the entire service.
3. Client side functional requirements: The use cases of the client side of the service are listed. For each use case, associated functional requirements are listed.
4. Server side functional requirements: The use cases of the server side of the service are listed. For each use case, associated functional requirements are listed.
5. Performance and usability requirements. For example: Mobile device memory limitation.
. Examples of usability and performance requirements are:
- Requirement 001: Midlets at runtime shall not require more than 662Kb. For mobile devices, runtime memory availability may be even lower.
- Requirement 002: A dedicated thread should be responsible for network access and management of the user, so the user can continue interacting with the device at all times.

2.8.2.3 Product Flow

This artifact is produced by the following activities:

- Select Requirements
- Write Scenarios
- Write Requests

This artifact is modified by the following activities.

- Requirements Phase
- Study Feasibility
- Specify Requirements
- Coding Phase

This artifact is used by the following activities:

- Search Possible Solutions
- Design Phase
- Design High Level
- Specify Conceptual Design
- Aquire COTS
- Identify COTS Products
- Evaluate COTS Products
- Select COTS Products
- Inspect Design
- Code
- Plan Testing Phase
- Plan Tests
- Testing Phase

2.8.3 Artifact: List Possible Solutions

2.8.3.1 Purpose

To hold the list of technical doubts found in the requirements specification and possible solutions for them.

2.8.3.2 Product Flow

This artifact is produced by the following activities:

- Search Possible Solutions

This artifact is not modified by any activity.
This artifact is used by the following activities:

- Test Possible Solutions

2.8.4 Artifact: *Feasibility Study*

2.8.4.1 Purpose

To hold the list of solutions that solve technological doubts found in the requirements specification and provide sound basis for accepting or rejecting these requirements.

2.8.4.2 Description

Structure of the document
SCOPE
The scope of this document is to describe the major responsibilities of the Server side and of the Client side for each service offered by the Pilot, in order to understand if the implementation is feasible in the defined timeframe.
BASIC FEATURES OF THE SERVICE
Describe the main characteristics of the service
CLIENT RESPONSABILITIES
List of the high-level client requirements

SERVER RESPONSABILITIES
List of the high-level server requirements
OBSERVATIONS
Performance requirements or constraints of the service
PROBLEMS
- Problem Id: Number
- Context: WISE Project Number of the pilot
- Description: Problem description
- Cause: Description of the problem?s reason.
- Solution (Reactive)
- Solution (Preventive)
- References: Links to other similar problems and solutions
- Additional documentation: Links to information that clarifies the technical background of the problem.

CONCLUSIONS
The feasibility study concludes whether the implementation of the service is feasible in the defined timeframe.
Guidelines could be extracted from the solutions of the problems.

2.8.4.3 Product Flow

This artifact is produced by the following activities:
- Test Possible Solutions

This artifact is modified by the following activities.
- Requirements Phase
- Study Feasibility

This artifact is used by the following activities:
- Specify Requirements
- Design Phase
- Design High Level
- Specify Conceptual Design
- Design Low Level

2.8.5 Artifact: *WISA Architectural Guidelines*

2.8.5.1 Purpose

To provide a unified and organized approach to the description of the software architecture.

2.8.5.2 Description

In particular, the stakeholders of this document are both technical and non-technical people. The former have to describe in detail those generic entities in terms of their implementing components. The latter have to understand the generic architectural entities making up a type of service.
The guidelines are basically made up of (1) a set of viewpoints to model the conceptual/concrete architecture (each describing a particular architectural aspect), and (2) the notation (i.e. languages and/or visual conventions) selected to model and represent each viewpoint. The views and diagrams in the architectural documents are based on these viewpoints and conform to the notation.

2.8.5.3 Product Flow

This artifact is not produced by any activity.
This artifact is not modified by any activity.
This artifact is used by the following activities:
- Design High Level
- Specify Conceptual Design
- Design Low Level

2.8.6 Artifact: *Architecture Document*

2.8.6.1 Purpose

To hold the high level (conceptual views) and low level (concrete views) design models that meet the specified requirements.

2.8.6.2 Description

The architecture is essentially a description of a software system, from several viewpoints. Its purpose is to address the concerns of the system stakeholders.
The description consists of several views, each conforming to a viewpoint. In practice the views are made up one or more models or diagrams
Structure of the document
1. Conceptual architecture
1.1 Conceptual structural view
1.1.1 System context (structure of the network)
1.1.2. Domain information models
1.1.3. Functional structure
1.2 Conceptual behavioural view
1.2.1 Collaboration diagram
1.3 Conceptual deployment view
1.3.1 Deployment diagram
1.4 Conceptual development view
1.4.1 Business model
1.4.2 Topology diagram
2. Concrete architechture
2.1 Structural view.
2.1.1 Information classes
2.1.2. Associations between classes
2.1.3 Essential computational-oriented aspects (Create inter component and intra component diagrams)
2.2 Concrete behavioural view
2.2.1 Inter component sequence diagrams
2.2.2 Intra component sequence diagrams
2.3 Concrete deployment view
2.3.1 Deployment diagram
2.4 Concrete development view
2.4.1 Table with the interfaces of components
2.4.2 Technology layers

2.8.6.3 Product Flow

This artifact is not produced by any activity.
This artifact is modified by the following activities.

- Design Phase
- Design High Level
- Specify Conceptual Design
- Design Low Level
- Coding Phase
- Code

This artifact is used by the following activities:

- Aquire COTS
- Identify COTS Products
- Evaluate COTS Products
- Select COTS Products

- Inspect Design
- Test Units
- Test Integration
- Testing Phase

2.8.7 Artifact: *Candidate COTS Products*

2.8.7.1 Purpose

To hold the list of possible components (COTS) that meet the architectureand their important information e.g., location, vendor, price.

2.8.7.2 Description

List of products together with a version that links to the site of the vendor.
If there are many versions of the products the version should be mentioned.
The list of products has the following structure.
- Product ID - Product name - Product version

2.8.7.3 Product Flow

This artifact is produced by the following activities:
- Identify COTS Products

This artifact is not modified by any activity.
This artifact is used by the following activities:
- Evaluate COTS Products
- Select COTS Products

2.8.8 Artifact: *Evaluation Results*

2.8.8.1 Purpose

To hold the list of possible components (COTS) that meet the architecture and their measured attributes.

2.8.8.2 Description

Document that contains the measures of all the attributes for all the products. It could be a basic Excel table.
For example:

- Product ID 1

Performance (attribute measure 1) = 5 out of 10
Portability (attribute measure 1) = 6 out of 10
Support (attribute measure 1) = 5 out of 10

- Product ID 2

Performance (attribute measure 1) = 9 out of 10
Portability (attribute measure 1) = 8 out of 10
Support (attribute measure 1) = 7 out of 10

2.8.8.3 Product Flow

This artifact is produced by the following activities:
- Evaluate COTS Products

This artifact is not modified by any activity.

This artifact is used by the following activities:
- Select COTS Products

2.8.9 Artifact: *Selected COTS Products*

2.8.9.1 Purpose

To hold the selected COTS products that meet the architecture

2.8.9.2 Description

This is a subset of the candidate COTS products. The list of selected COTS products has the following structure:
- Product ID - Product name - Product version

2.8.9.3 Product Flow

This artifact is produced by the following activities:
- Aquire COTS
- Select COTS Products

This artifact is not modified by any activity.
This artifact is used by the following activities:
- Specify Conceptual Design

2.8.10 Artifact: *Inspection Log*

2.8.10.1 Purpose

To gather the problems found in the architecture document, and the suggestions to overcome them.

2.8.10.2 Description

Structure of the document
Date: Date of the inspection
Team members: Names of the inspection participants
Problem number: Identifier for the problem found
Problem description: Description of the problem found
Suggestion: Description of the suggestions to solve the problem

2.8.10.3 Product Flow

This artifact is produced by the following activities:
- Inspect Design

This artifact is modified by the following activities.
- Design Phase

This artifact is used by the following activities:
- Design High Level
- Specify Conceptual Design
- Design Low Level

2.8.11 Artifact: *Design Inspection Checklist*

2.8.11.1 Purpose

To hold the items to be checked from the Architecture Document.

2.8.11.2 Description

Structure of the document
A simple table with the following contents:
All Design Diagrams
Is each name unique?
Are all names used in the diagrams consistent?
Are all names used in the diagrams correct?
Do the design diagrams cover the system requirements?
Check
Component diagrams (Intra-Inter)
Does the component represent all interchangeable parts of the system?
Are the interfaces of all components defined?
Are the interfaces of all components correct?
Can (and should) interfaces be simplified?
Check
Class diagrams
Are the classes consistently documented?
Does each class denote a collection of similar instances?
Check
Attributes
Does each class attribute in the design class diagram have an associated data type?
Are all data types primitives? If the data type is not primitive, could an association to an existing class replace it?
Have initial values been specified for attributes?
Check
Associations
Is the cardinality of each association correct?
Are role names given for each of the classes involved in a recursive association?
Are role names given for classes that have more than one association?
Are role names consistent?
Does each association require a persistent representation? If not, could it be better modeled as an operation?
Could an aggregate be better modeled with attributes?
If the set of sibling classes differs only in the value of one attribute, could it be changed to an enumerated attribute in the parent class?
Sequence diagrams (intra-inter)
Is the sequence diagram consistent with the classes diagram?
Do the objects used in the sequence diagram belong to a class defined in the classes diagram?
Are all use cases mapped to a sequence diagram?
Are all variations of use cases modeled?
Are the receiver objects available?
Check
Deployment diagrams
Are all the concrete components mapped to the nodes on the execution environment?
Are the business model roles associated to the concrete components?
Are all the relationships between the components modeled?
Check
Technology platform diagrams
Were the native platform services identified?
Were the vendor specific database services identified?
Were the (server-client) services identified?

2.8.11.3 Product Flow

This artifact is not produced by any activity.
This artifact is not modified by any activity.
This artifact is used by the following activities:

- Inspect Design

2.8.12 Artifact: *Source Code*

2.8.12.1 Purpose

To hold the lines of code that implement the design.

2.8.12.2 Description

Lines of code written that conform to the coding standard.

2.8.12.3 Product Flow

This artifact is produced by the following activities:

- Coding Phase
- Code

This artifact is modified by the following activities.

- Test Units
- Test Integration
- Test System

This artifact is used by the following activities:

- Testing Phase

2.8.13 Artifact: *Integrated Code*

2.8.13.1 Purpose

To hold the lines of code that implement the design and meets the requirements specification.

2.8.13.2 Description

Integrated and tested units of code that conforms to the coding standard

2.8.13.3 Product Flow

This artifact is produced by the following activities:

- Test Integration

This artifact is modified by the following activities.

- Release Code

This artifact is used by the following activities:

- Test System

2.8.14 Artifact: *Integration Report*

2.8.14.1 Purpose

To hold the results of the integration of the product.

2.8.14.2 Description

Structure of the document

1. Defect id
2. Defect description
3. List of activities to fix the defect
4. Products to be modified (Requirements, design, code, etc....)

2.8.14.3 Product Flow

This artifact is not produced by any activity.
This artifact is modified by the following activities.

- Test Integration

This artifact is not used by any activity.

2.8.15 Artifact: *Test Plan*

2.8.15.1 Purpose

To hold the set of test cases that validate each of the specified requirements.

2.8.15.2 Description

Structure of the document
List of test cases. Each test case contains the following fields:
- Date
- Authors
- Team/Project
- The factors to be tested: functional testing, performance testing, or usability testing
- Data needed
- Hardware needed
- The procedure to set the test
- The requirement to be tested
- The supporting materials required for each test
- The description of the test
- References to other test cases

2.8.15.3 Product Flow

This artifact is produced by the following activities:

- Plan Testing Phase
- Plan Tests

This artifact is not modified by any activity.
This artifact is used by the following activities:

- Build Test Framework
- Testing Phase
- Test System

2.8.16 Artifact: *Test Framework*

2.8.16.1 Purpose

To support the execution of test plans.

2.8.16.2 Description

The adequate parts for testing wireless Internet services as e.g., wireless network, server, mobile devices, emulators, simulators, and their interconnections constitute the test framework.

2.8.16.3 Product Flow

This artifact is produced by the following activities:
- Plan Testing Phase
- Build Test Framework

This artifact is modified by the following activities.
- Plan Tests
- Test System

This artifact is used by the following activities:
- Testing Phase

2.8.17 Artifact: *Product Tested*

2.8.17.1 Purpose

To hold the integrated and tested product.

2.8.17.2 Description

Integrated and tested system ready to be released to the end customers.

2.8.17.3 Product Flow

This artifact is produced by the following activities:
- Testing Phase
- Test System

This artifact is not modified by any activity.
This artifact is used by the following activities:
- Test Acceptance
- Test Usability

2.8.18 Artifact: *Test Report*

2.8.18.1 Purpose

To hold the most relevant information of the system test.

2.8.18.2 Description

Structure of the document
Header
1. The date the system test was run
2. The name of the person running the system test
Body
3. The test plan's name and number
4. The product tested
5. The number of defects found
6. The test results

2.8.18.3 Product Flow

This artifact is not produced by any activity.
This artifact is modified by the following activities.
- Testing Phase
- Test System

This artifact is used by the following activities:
- Analyze Defect

2.8.19 Artifact: *Usability Interview*

2.8.19.1 Purpose

To provide guidelines to interview users and find usability problems.

2.8.19.2 Description

This document contains the questions to be followed by the users in order to evaluate the usability of the application.
The structure is as follows:
- Scope
- Questions for Evaluating Web Sites
- Questions for Online Communities
- Questions for Video Games
- Problems and Suggestions

2.8.19.3 Product Flow

This artifact is not produced by any activity.
This artifact is not modified by any activity.
This artifact is used by the following activities:
- Test Usability

2.8.20 Artifact: *Usability Report*

2.8.20.1 Purpose

To describe problems or suggestions found related with the usability of the service.

2.8.20.2 Description

This document contains the summary of the problems found in the usability test, and the suggestions made in order to improve the product with their respective priority.
The structure is:
- SCOPE
- List of problems/suggestions order by priority

2.8.20.3 Product Flow

This artifact is produced by the following activities:
- Test Usability

This artifact is not modified by any activity.
This artifact is used by the following activities:
- Analyze Defect

2.8.21 Artifact: *Acceptance Test Report*

2.8.21.1 Purpose

To hold problems or suggestions found during the tests of the service.

2.8.21.2 Description

Structure of the document
1. The date the acceptance test was run
2. The name of the person running the acceptance test

3. The product tested
4. The description of defects found
5. Suggestions

2.8.21.3 Product Flow

This artifact is produced by the following activities:
- Test Acceptance

This artifact is not modified by any activity.
This artifact is used by the following activities:
- Analyze Defect

2.8.22 Artifact: *Analyze Defect Report*

2.8.22.1 Purpose

To hold the inmediate plans to handle the results of testing the system.

2.8.22.2 Description

Structure of the document
1. Defect id
2. Defect description
3. List of activities to fix the defect
4. Products to be modified (Requirements, design, code, etc....)

2.8.22.3 Product Flow

This artifact is produced by the following activities:
- Analyze Defect

This artifact is not modified by any activity.
This artifact is not used by any activity

2.9 *WISEP* / *ROLES*

2.9.1 Role: *Customer*

2.9.1.1 Description

This role is assumed by individuals with knowledge and experience in designing and creating new services and products for the mobile devices of the company.

2.9.1.2 Participation

This role participates in the following activities:
- Requirements Phase
- Select Requirements
- Test Acceptance
- Test Usability

2.9.2 Role: *Developer*

2.9.2.1 Description

This role is assumed by the members of the project with knowledge and experience in programming Wireless Internet Applications.
They are in charge of developing the service across all of its lifecycle. They take part in estimating the effort, analyzing requirements, designing the service, implementing and testing it.
They all may share the task of configuring the various environments and managing the configuration. They are assigned to different tasks according to planning done by the Project Leader and Project Manager. The required skills for this domain are:
- Programming skills / Project support skills
- Knowledge of the wireless environment
- Network programming and configuration or embedded software experience are preferred.

2.9.2.2 Participation

This role participates in the following activities:

- Requirements Phase
- Write Scenarios
- Write Requests
- Study Feasibility
- Search Possible Solutions
- Test Possible Solutions
- Specify Requirements
- Design Phase
- Design High Level
- Specify Conceptual Design
- Aquire COTS
- Identify COTS Products
- Evaluate COTS Products
- Select COTS Products
- Design Low Level
- Inspect Design
- Coding Phase
- Code
- Test Units
- Test Integration
- Release Code
- Plan Testing Phase
- Plan Tests
- Build Test Framework
- Testing Phase
- Test System
- Test Usability
- Analyze Defect

2.9.3 Role: *Project Leader*

2.9.3.1 Description

The members of the project with knowledge and experience in programming Wireless Internet Applications, and experience as managers of software development projects assume this role.
The project leader is in charge of the relationships with the client and elicits the requirements from the client.

He is in charge of managing the project and the project plan. The project plan is usually filled in by the Project Manager.
He is in charge of monitoring and controlling the project.
He is in charge of evaluating the risks and issues and finding out how to solve them.
He is a team member, so in the absence of urgent issues he takes part in the development of the project.

2.9.3.2 Participation

This role participates in the following activities:

- Requirements Phase
- Select Requirements
- Specify Requirements
- Design Phase
- Inspect Design
- Coding Phase
- Release Code

2.9.4 Role: *Project Manager*

2.9.4.1 Description

Manages people and resources. Gets the project and is the one generally responsible for it.
He is responsible for managing costs and staffing.
He is responsible for the team members' working time (he authorizes extra working days and overtime work together with the senior management).
He is responsible for negotiating with the client the costs and the staffing for the project and, in case of major issues, to re-negotiate them.

2.9.5 Role: *Technical Leader*

2.9.5.1 Description

Leads the technical issues.
He is responsible for the design documents. He approves them together with the project leader.
He reviews the SRS and gives the general guidelines for the design.
He knows the technology and clarifies developers' technical questions.
He is capable of following the different tasks in which the developers are involved and giving advice to them.
He masters the situation of the development and reports the risks to the project leader.
In case of big projects, several technical leaders can manage different technical areas of the project.
He is a team member, so in the absence of urgent issues he takes part in the development of the project.

2.9.5.2 Participation

This role participates in the following activities:

- Analyze UI Feasibility
- Select Feasible Requests
- Design Phase
- Design High Level
- Design Low Level
- Inspect Design
- Coding Phase
- Test Integration
- Release Code
- Plan Testing Phase

- Plan Tests
- Testing Phase
- Test System
- Analyze Defect

2.10 *WISEP / TOOLS*

2.10.1 *Tool: UML Editor*

2.10.1.1 Description

Tool for creating and maintaining documents whose contents are standard UML diagrams.

2.10.1.2 Usage

This tool is used by the following activities:

- Write Scenarios
- Design Phase
- Design High Level
- Specify Conceptual Design
- Design Low Level
- Inspect Design

2.10.2 Tool: *Integrated Development Environment*

2.10.2.1 Description

An integrated development environment (IDE) is a programming environment that has been packaged as an application program, typically consisting of a code editor, a compiler, a debugger, and a graphical user interface (GUI) builder. The IDE may be a stand-alone application or may be included as part of one or more existing and compatible applications.
Important integrated environments for Java are:
Visual Age for Java
JBuilder
JDK
Sun Studio: This environment was improved for using J2ME.

2.10.2.2 Usage

This tool is used by the following activities:

- Requirements Phase
- Study Feasibility
- Test Possible Solutions
- Aquire COTS
- Evaluate COTS Products
- Coding Phase
- Code
- Test Units
- Test Integration
- Release Code
- Plan Testing Phase
- Build Test Framework
- Testing Phase
- Test System
- Test Acceptance

- Analyze Defect

2.10.3 Tool: *Real Mobile Device*

2.10.3.1 Description

Mobile devices with support for 2G, 3G wireless networks are:
- Desktops/Notebooks
- Tablet/PC
- PDA
- Handheld PC
- PDA/Phones
- Smart /Phones
- Cellular/Phones

2.10.3.2 Usage

This tool is used by the following activities:
- Requirements Phase
- Study Feasibility
- Test Possible Solutions
- Coding Phase
- Test Units
- Test Integration
- Plan Testing Phase
- Build Test Framework
- Testing Phase
- Test System
- Test Acceptance
- Test Usability
- Analyze Defect

2.10.4 Tool: *Text Editor*

2.10.4.1 Usage

This tool is used by the following activities:
- Requirements Phase
- Select Requirements
- Analyze UI Feasibility
- Write Requests
- Select Feasible Requests
- Specify Requirements
- Design Phase
- Design High Level
- Specify Conceptual Design
- Evaluate COTS Products
- Select COTS Products
- Design Low Level
- Inspect Design
- Plan Testing Phase
- Plan Tests
- Testing Phase

REFERENCES

[1] Buttler, M.H.: Current Technologies for Device Independence. HP Laboratories, HP-2001-83, Bristol., (2001).
[2] http://www.w3.org/TR/2001/WD-di-princ-20010918/
[3] Forchino, F., Negro, M.: Management of Heterogeneous Clients. Deliverable D3. Wireless Internet Service Engineering. IST-2000-30028.
[4] Ritel, H., Webber, M.: Planning problems are wicked problems. In: Cross N (ed). Developments in Design Methodology. Wiley, Chichester, pp 135-144, (1984).
[5] Beck, K.: Extreme Programming Explained: Embrace Change. Addison Wesley, (2000).
[6] Highsmith, J.: What is Agile Software Development? The Journal of Defense Software Engineering. October, (2002).
[7] Boehm, B.W.: A Spiral Model for Software Development and Enhancement, IEEE Computer, vol 21, No 5, pp. 61-72 (1988).
[8] Karlsson, E.: A Construction Planning Process. Q-Labs, LD/QLS 96:0381, Lund Sweden (1999).
[9] Kovari, P., Acker, V.B., Marino, A., Ryan, J., Tang, L.K., Weiss, C.: Mobile Applications with Websphere Everyplace Access Design and Development. IBM SG24-6259-00 (2001).
[10] Taylor, M.J., McWilliam, J., Forsyth, H., Wade, S.: Methodologies and Website Development: A Survey of Practice. Information and Software Technology, vol. 44, No. 6, pp. 381-391 (2002).
[11] Giannetti, F.: Device Independency Web Application Framework. In: W3C Device Independent Authoring Techniques Workshop, 25-26 (2002).
[12] Wong, C., Chu, H., Katagiri M: A Single Authoring Technique for Building Device Independent Presentations. In: W3C Device Independent Au-thoring Techniques Workshop, (2002).
[13] Wong, C., Chu, H., Katagiri, M.: GUI Migration Across Heterogeneous Java Profiles. Proceedings of ACM SIGCHI-NZ?02. (2002).
[14] Mori, G., Paterno, F., Santono, C.: An XML Based Approach for Design-ing Nomadic Applications. In: W3C Device Independent Authoring Techniques Workshop, (2002).
[15] Roe, C., Gonik, S.: Server-Side Design Principles for Scalable Internet Systems. IEEE Software, vol.19, No. 2, pp. 34-41 (2002).
[16] Friday, A., Davies, N., Blair, G.S., Cheverest, K.W.J.: Developing Adap-tive Applications: The MOST Experience. Integrated Computer Aided Engineering: ICAE, vol 6, No 2, pp. 143-158 (1999).
[17] Cushnie, J., Hutchison, D., Oliver, H.: Evolution of Charging and Billing Models for GSM and Future Mobile Internet Services. Lecture Notes in Computer Science, vol. 1922, pp. 312-323 (2000).
[18] Odlyzko, A.: Paris Metro Pricing: The Minimalist Differentiated Ser-vices Solution. AT&T Laboratories Research, April 1999.
[19] Nerurkar, U.: Web User Interface Design: Forgotten Lessons. IEEE Software, vol. 18, No. 6, pp. 69-71 (2001).
[20] Rosenfeld, L., Morville, P.: Information Architecture for the World Wide Web, O?Reilly, New York, (1998).
[21] http://java.sun.com/j2me/docs/alt-html/midp-style-guide7/preface.html
[22] Kalaoja, J., Niemelä, E., Tikkala, A., Kallio, P., Ihme, T., Torchiano, M.: WISA reference architecture, WISA architectural knowledge base (WISA) and Reference Architecture (WISA/RA), Deliverable D4 (Parta).
[23] Maiden, N., Ncube, C.: Acquiring COTS Software Selection Requirements. IEEE Software: 46-56, March/April (1998).
[24] Kontio, J.: A Case Study in Applying a Systematic Method for COTS Selection. Proc. of IEEE-ACM 18th International Conference on Software Engineering, pp. 201-209, (1996).
[25] Ochs, M.A., Pfahl, D., Chrobok-Diening, G.: A Method for Efficient Measurement-based COTS Assessment ad Selection - Method Description and Evaluation Results. Proc. of IEEE 7th International Software Metrics Symposium, pp. 285-296. (2001).
[26] Morisio, M., Tsoukiàs, A.: IusWare: A methodology for the evaluation and selection of software products, IEE Proceedings-Software, 144 (3): 162-174. (1997).
[27] Feblowitz, D.M., Greenspan, S. J.: Scenario-Based Analysis of COTS Acquisition Impacts. Requirements Engineering, 3 (3/4): 182-201 (1998).

[28] Lawlis, P.K., Thomas, D.M., Courtheyn,T.: A Formal Process for Evaluating COTS Software Products, IEEE Computer, 34 (5): 58-63, (2001).
[29] Comella-Dorda, S., Dean, J., Morris, E. Oberndorf, T.: A Process for COTS Software Product Evaluation. Proc. of 1st International Conference on COTS Based Software Systems (ICCBSS), Orlando (FL), February 4-6, pp. 86-96. (2002).
[30] Humprey, W.S.: Introduction to the Team Software Process. SEI Series In Software Engineering. Adisson Wesley. (1999).
[31] Read, K., Maurer, F.: Developing Mobile Wireless Applications. IEEE Internet Computing, vol 7, No 1, pp. 81-86. (2003).
[32] Colberg, C.S., Thomborson, C.: Watermarking, Tamperproofing and Obfuscation- Tools for Software Protection. IEEE Transactions on Software Engineering, vol. 28, No. 8, pp. 735-746, (2002).
[33] Maurer, F., Martel, S.: Rapid Development for Web-Based Applications. IEEE Internet Computing, vol 6, No 1, pp. 86-90 (2002).
[34] Kohler, E., Morris, R., Chen, B., Jannotti, J., Kaashoek M. F.: The Click Modular Router, proceedings of ACM Transactions on Computer Systems, vol. 18, pp. 263-297, (2000).
[35] Rizzo, L.: Dummynet: a simple approach to the evaluation of network protocols, proceedings of ACM Computer Communication Review, vol. 27, (1997).
[36] Fall, K.: Network Emulation in the VINT/NS Simulator, In Proceedings of 4th IEEE Symposium on Computers and Communications, (1999).
[37] Noble, B.D., Satyanarayanan, M., Giao, T.N, Katz, H.R.: Trace-Based Mobile Network Emulation, Proceedings of the ACM SIGCOMM '97 conference on Applications, technologies, architectures, and protocols for computer communication.
[38] Satoh, I.: Flying Emulator: Rapid Building and Testing of Networked Applications for Mobile Computers, Proceedings of 5th International Conference on Mobile Agents (MA'2001), Lecture Notes in Computer Science (LNCS), vol. 2240, pp.103-118, Springer, (2001).
[39] Zheng, P, Ni, M.N: EMPOWER: A Network Emulator for Wireless and Wireline Networks. In Proceedings of IEEE INFOCOM (2003).
[40] El-Far, I.K., Thompson, H,H., Mottay, F.E.: Experiences in Testing Pocket PC Applications. In Proceedings of the Fifth International Software and Internet Quality Week Europe, (2002).
[41] Puuskari, M:, General Packet Radio Services for GSM, University of Helsinki Department of Computer Science Spring (2000).
[42] Jain, A, Tyago, S.: Simulation Testing in Mobile Networks ? Protocols and Applications Perspective, Telecom & Networking Division, HCL Technologies Ltd.
[43] Thimbleby H., Cairns, P., Jones M.: Usability Analysis with Markov Models, ACM Transactions on Computer-Human Interaction, Vol. 8, (2001).
[44] Chi, E.: Improving Web Usability Through Visualization. IEEE Internet Computing, vol. 6, No 2, pp. 64-71 (2002).
[45] Upchurch, L., Rugg, G., Kitchenham, B.: Using Card Sorts to Elicit Web Page Quality Attributes. IEEE Software, vol. 18, No. 4, pp. 84-89 (2002).
[46] Becker, A.S.: A Usability Perspective on Wireless Internet Technology, Computer Science & Software Engineering, Florida Institute of Technology, (2001).
[47] Sinha, A., Chandrakasan, A.: JouleTrack - A Web Based Tool for Software Energy Profiling, Proceedings of the 38th Design Automation Conference, (2001).
[48] Nielsen, J., Mack, R.L.: Usability Inspection Methods / John Wiley & Sons, Inc; (1994).