		Title: Service Engineering Process (Empirical approach for creating the reference process model)
		Version: 03.05 Date : 17 Sep 04 Pages : 47
		Author(s): Alexis Ocampo, Jürgen Münch
		To: WISE CONSORTIUM
The WISE Consortium consists of: Investnet, Motorola Technology Center Italy, Sodalía s.p.A, Sonera, Solid EMEA North, Fraunhofer IESE, Politecnico di Torino, VTT Electronics		Printed on: 17/09/2004 10.18
Status: <input type="checkbox"/> Draft <input type="checkbox"/> To be reviewed <input type="checkbox"/> Proposal <input checked="" type="checkbox"/> Final/Released	Confidentiality: <input checked="" type="checkbox"/> Public - Intended for public use <input type="checkbox"/> Restricted - Intended for WISE consortium only <input type="checkbox"/> Confidential - Intended for individual partner only	
Deliverable ID: D2 (Part A)		
Title: Service Engineering Process (Empirical approach for creating 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 approach developed and followed in order to develop the software process reference model: The first chapter describes the motivation of the work and the goals, and the strategies taken to accomplish the goals are presented. As part of the strategy, the survey of related work and its results are presented in the second chapter. Chapter 3 presents the mechanisms used to elicit process knowledge for the development of descriptive process models, to analyze the commonalities and differences between processes, and to create the reference process model. Chapter 4 briefly presents the relationship between the process models and the measurement program (see deliverables D8.v2 and D9.v2). Chapter 5 presents the activities that were enacted during the second iteration with the objective of enhancing the quality of the products. Chapter 6 presents a summary of the presented work. Appendix 1 gives a characterization schema for the pilot projects and their context.		



	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 2 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

TABLE OF CONTENTS

1. Introduction	4
1.1 Motivation.....	4
1.2 Goal	5
1.3 Strategy.....	5
1.4 Documentation	6
1.5 Definition of a Process to Engineer Services	7
1.6 Structure of the Deliverable	11
2. Survey Processes for Software and System Engineering	12
2.1 Definition of the Literature Search Scope	12
2.2 Definition of Information Sources.....	12
2.3 Analysis and Classification of the Results.....	13
2.3.1 Life Cycle Processes	13
2.3.2 Requirements	14
2.3.3 Design	16
2.3.4 Implementation	19
2.3.5 Test.....	20
2.3.6 Managerial Processes.....	23
2.4 Summary of Practices	23
3. Define New Process to Develop Wireless Services	25
3.1 Elicit Existing Process Knowledge	25
3.1.1 Process Models	26
3.2 Analyze Models	27
3.3 Reference Process Model	30
3.3.1 Reference process model creation.....	30
4. Define Measures and Indicators.....	30
5. Evaluate Quality-Related Activities	31
5.1 Pilot 1	31
5.1.1 Verification Activities	31
5.1.2 Validation Activities	31
5.2 Pilot 2	31
5.2.1 Verification Activities	31
5.2.2 Validation Activities	33
6. Summary and Results.....	34
Appendix 1. Characterization Vector Definition.....	36
Categories	37
Domain Characteristics	37
Implementation Characteristics	38
Enterprise characteristics.....	42
References.....	44


	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 3 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

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

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 4 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

1. INTRODUCTION

1.1 MOTIVATION

Experience indicates that developing software with high quality requirements can be done successfully if an explicitly defined process is followed. Furthermore, lack of a development process makes accurate planning very difficult and in many cases impossible. Experience from progressive software development organizations like the NASA Software Engineering Laboratory (SEL) [14], for instance, has shown that one essential precondition for developing software of a predetermined quality in a predictable fashion is the design, establishment, and use of systematic software development processes.

Process deployment can fail especially if an organization does not put enough emphasis into the design and promotion of process models and the infrastructure needed for process deployment. Early results from a multi-case study conducted at Nokia Mobile Phones clearly show the importance of a stable implemented infrastructure for process deployment [7]. A prerequisite for this are explicitly defined process models for the application domain that are tailorable to specific project contexts.


New and unknown application domains do not have explicitly defined software development processes. Furthermore, the design and introduction of such processes is very risky, because typically, there exists no previous experience on which processes or process fragments are suitable and executable in the environment of the developing organization.

An application domain that has to deal especially with such problems is the Wireless Internet Services domain because its development cycles are very short. In order to produce software of sufficient quality and thus remain competitive in the market, an appropriate and piloted development process is needed very quickly. This is valid in general for a new domain, but it is especially valid for the Wireless Internet domain: If a specific process for Wireless Internet Services is not defined, the risk exists that the process followed in Internet Services development will also be inherited for Wireless Internet Services. As the Wireless Internet gets popular, the Internet Service providers will try to provide the same services over the Wireless Internet as well, and they may easily try to follow the same development process they use for Internet Services. This is very risky because the wireless world is different from the fixed world and additional issues must be considered during the implementation of services in order to get a final product with a certain level of quality, which can be competitive on the market.

How could these risks be prevented or at least minimized? Looking carefully at software organizations while they apply their technology knowledge on a realistic setting, and also learning from documented experiences can provide valuable input and deep understanding of the Wireless Internet Services domain.

The WISE project follows an underlying experimental paradigm: Experimenting methodology and technology in real life applications is seen as the key to understanding, validating and improving methodology and technology. Therefore, several pilot developments have already been performed or are planned in the near future.

The work described in this deliverable was conducted in the context of the WISE project (Wireless Internet Software Engineering), which was started in 2001 and run until 2004. The project aims at delivering methodologies and technologies to develop services on the Wireless Internet. The methodology part comprises an overall process to drive the engineering of mobile services, a business model to specify roles and skills of involved parties, and guidelines to handle heterogeneous clients (e.g., handhelds, laptops). The technology part comprises a high-level architecture for mobile services, a service management component, a data replication and synchronization component, and software agents to support negotiation functions in components.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 5 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

This deliverable describes the results with respect to the software process for the **third and final iteration** of the pilot projects. Industrial partners responsible for the pilot development and the underlying infrastructure are Investnet (Pilot 1), Motorola Global Software Group – Italy, and VTT (Pilot 2).

The industrial partners mentioned above identified several success factors for wireless Internet services, especially time-to-market, the ability to quickly deliver functionality with simultaneous fulfilment of high quality requirements, and high usability requirements in terms of service performance. These quality requirements vary with different services. For example, Wireless On-line Trading Services (Pilot 1) require particularly high reliability, functional correctness, scalability, and redundancy. On the other hand, Wireless On-line Entertainment Services (Pilot 2) has no strict reliability requirements, but it has even stricter requirements in terms of usability and performances of the designed service architecture.

Research partners responsible for developing processes, methods and tools are the Fraunhofer Institute for Experimental Software Engineering IESE (Germany), Politecnico di Torino (Italy), and VTT Electronics (Finland). The Fraunhofer Institute IESE is responsible for the modeling of the Service Engineering Processes (Task 1.2), which is documented in this deliverable.

1.2 GOAL

The general objective of WP1 is to define a comprehensive methodology to engineer and operate Wireless Internet Services. This objective has been split up into the following tasks: Task 1.1 Business model, Task 1.2 Process and Task 1.3 Heterogeneous clients.

The proposed steps to accomplish task 1.2 are:

- Survey processes for software and system engineering.
- Evaluate formalisms to describe software processes, and select one.
- Elicit existing process knowledge.
- Define new process to develop wireless services, in terms of activities, techniques, tools, deliverables, and milestones.
- Define measures and indicators to control the cost, time and quality of the process and its deliverables.
- Evaluate quality-related activities and techniques (testing, inspections, reviews, prototypes) and indicators, adapt and include them into the process.

1.3 STRATEGY

Figure 1-1 shows the planned iterations to be performed during the WISE project in order to accomplish the WP1 goals. A first iteration started with an outline of an initial process as performed in the pilot's organizations. These process models served, on the one hand, as a basis for further iterations, on the other hand, the initial process models increased the understanding of the application domain. Then in the second iteration the process models were integrated into one together with the techniques and practices discovered during the iteration 1. Finally, in the third iteration, enhanced practices were integrated in a tailorable process.

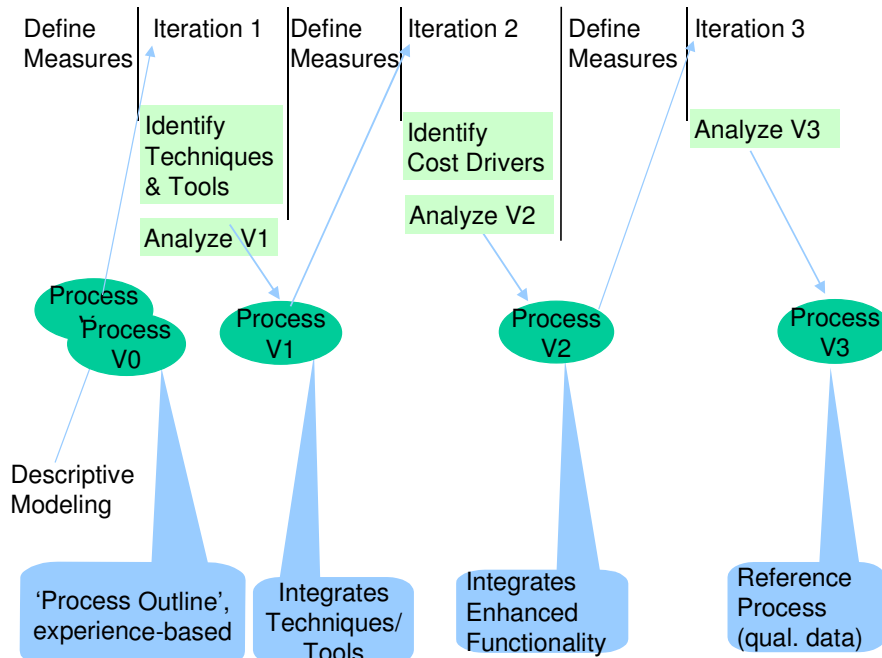



Figure 1-1: Planned process model iterations in the WISE project

1.4 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 A 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)

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 7 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

1.5 DEFINITION OF A PROCESS TO ENGINEER SERVICES

An explicit process model is a key ingredient for high productivity and software quality. Since software development projects are unique regarding their combination of specific goals and characteristics, providing 'ideal' and, at the same time, universal development processes is not a suitable solution for real life [76]. Instead, effective and efficient software development processes tailored to the particularities of the application domain and project constraints are required.

The wireless Internet services domain is an upcoming new application domain, which can be characterized as follows: quickly evolving technology, upcoming new devices, new communication protocols, support for new different media types, varying and limited communication bandwidth, together with the need for new business models that will fit in with the completely new services portfolio. Examples of new Wireless Internet Services can be expected in the domain of Mobile Entertainment, Telemedicine, Travel Services, Tracking and Monitoring Services, or mobile trading services. At the moment, there is very little experience in developing software for such services systematically. Therefore, designing processes for this domain implicates several difficulties:

1. Whereas for conventional software development, several standards exist, for Wireless Internet Services no such standards are available that could be used as reference.
2. The Wireless Internet Services domain lacks specific experience on particular techniques, their applicability and constraints.
3. The variations of the applications and, as a consequence, possible variations of the development processes, are not sufficiently understood.
4. The impact of the variation of the enabling technology on the developed service is not always known and this may affect the development process.

There are several ways towards solving this problem: one widely accepted idea in the software engineering community is descriptive modeling of development processes, which leads to the explicit definition of process models, product models, and resource models [18].

Descriptive software process modeling captures processes as they take place in development. Therefore, the initial process model elicited from a software organization that faces a new project on Wireless Internet Services is a practice adopted from a similar domain. Then, establishing baselines (e.g., an effort baseline), and collecting and using measurement data may further enhance the understanding and control of software development processes, products, and relationships between them [65].

From the viewpoint of a Process Engineer, the following questions arise:

- How can we quickly adapt software development processes from other domains for the development of Wireless Internet services?
- How can processes be speed up by perpetuating acceptable quality?
- Which existing techniques, methods and tools can be used?
- How should these be selected, adapted, and integrated into the process?
- What are typical variations of the processes in this domain?
- What are the impact factors on the effects of the processes?
- What kind of documentation is required?

In the context of the WISE project IESE has developed a new method in order to aggregate the activities planned for the process model iterations, and to empirically design development processes for new domains (i.e., reference processes [29]). The overall method can be applied to unknown new domains in general, but as the focus of this work is the Wireless Internet domain, special emphasis is placed on the particularities of this domain.

The goal of the method is to rapidly come up with a process that considers existing experience. The process is subsequently evaluated in pilot projects. As a consequence, drastic risk reductions in developing applications are expected.

The two key ingredients for the method are the set-up of selected pilot projects and the creation of descriptive process models from the pilot projects. The pilot projects ought to rely as much as possible on practices already in place in the development organization. For instance, new domains may require new practices or adaptations of existing practices. Variations and commonalities of processes need to be identified. Commonalities may indicate typical process steps and can lead to abstractions of the process in the model; variations are indicators for possible factors impacting the process and may lead to specializations of the process. Finally, the process models are integrated to form a comprehensive process model.

The overall method, its steps along with the major inputs, and output products are depicted in the following figure.

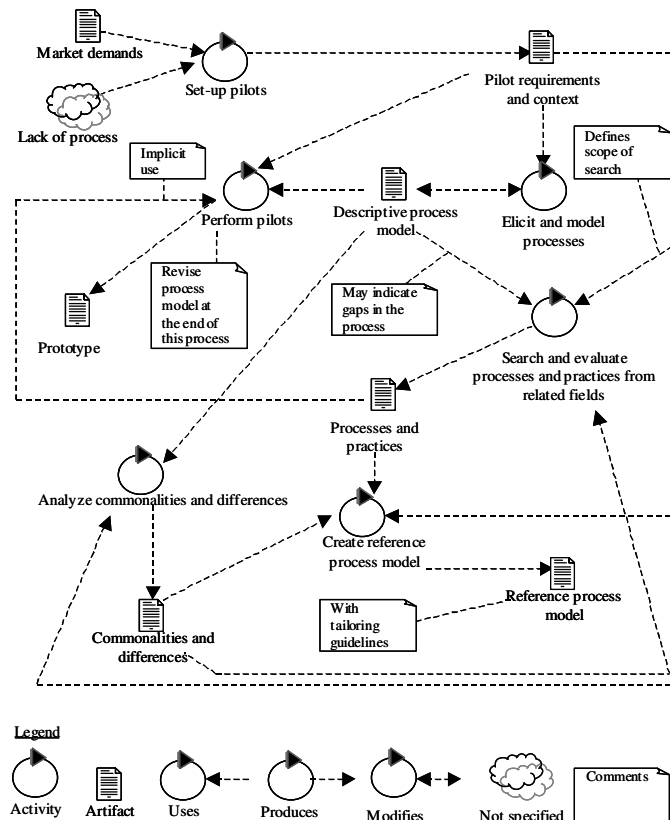



Figure 1-2: Top level activities of the method

The method consists of the following steps: In the first step, *set-up pilots*, suitable pilot projects have to be determined and organized. Pilot projects are to be determined by market demands in such a way that the pilots are representative for the new application domain. In the step *perform pilots*, the pilot projects are executed. In the step *elicit and model processes*, the processes as performed in the pilot projects are observed and modeled, resulting in a set of descriptive process models. A first version of the process models can be obtained based on similar past projects. The corresponding information is obtained through interviews with involved persons and other information sources, such as project plans or process artifacts. In parallel to these three steps, a step *search and evaluate processes and practices from related fields* is performed: The process engineer looks for processes and practices from related areas. This information will be used to fill the process model where it is incomplete and to introduce new practices into the process where old practices were seen as inefficient or are no longer adequate. In the step *analyze commonalities*

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 9 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

and differences, commonalities and differences between the different process models have to be analyzed in order to identify process variants and justifications for them. This must recognize differences in the application domain as well as goals and contexts of the pilot developments. In the final step, *create reference process model* (in the previous versions *create comprehensive process model*), the descriptive models for the pilots, practices and processes from related fields are integrated into a reference process model. Accompanying these steps, continuous improvement of the process during the pilot development with continuous flow of feedback will help to tailor the process during development and identify necessary changes early on.

This approach has several benefits: first, performing pilot projects and modeling their processes reveals the strengths and weaknesses of the processes early on. This can be seen as process prototyping. For an organization that introduces a process designed in such a descriptive manner, this reduces potential risks related to the introduction of a newly designed process. Second, introducing a new process based on existing practices typically requires a smaller shift in work procedures and is therefore more likely to be accepted by the process performers. Third, this concept allows for an incremental approach, which is more manageable than introducing a process in one shot. An additional benefit of this approach is that process performers of the domain are directly involved and can contribute to the development of the new process. Therefore, the process is more likely to be accepted and adapted. A bottom-up approach allows to quickly get an accurate model and to avoid problems with theoretical models that do not fit and that are not adequately tailored.

Set-up pilots

In order to better understand the processes as well as the application, several pilots should be set up. Looking at future market requests and what is new and interesting to be investigated by a pilot is a major impact factor on the specification pilot contexts and goals. Additionally, the new domain has to be characterized in order to search for related projects and experience in the developing organization. This characterization is documented within the target context. Afterwards, similar projects are searched and assessed with respect to the reuse potential of practices (i.e., techniques, methods, tools) and processes for the new domain. The result is a set of selected projects.


Based on the identified market demands and the experience from the selected projects, the requirements for the pilots and their specific contexts are defined. In addition, the pilots have to be planned and organized. Based on market demands (such as the need to adapt existing services for the Internet towards Wireless Internet services or to create new services) and companies' interests, two target contexts for the two pilots were defined: initially the development of a Wireless Internet Service for mobile online trading (Pilot 1) and the development of a service for mobile entertainment (Pilot 2).

The goal of Pilot 1 is to provide a service for the management of a virtual portfolio. For Pilot 1, similar projects could be identified that are concerned with the development of Internet trading services (i.e., development of a market informational and trading simulator site). Pilot 1 is an adaptation of this service to the wireless domain. The requirements for Pilot 1 comprise very high availability, correctness of data, and stringent reliability of customer identification and authorization as well as instantaneous response time in terms of quick data provision. The goal of Pilot 2 is the development of a multiplayer online game operated from mobile terminals. The requirements for Pilot 2 comprise the ability for user interaction on a shared environment, short response times and portability to different platforms.

The description of the pilot contexts is defined through a characterization vector (see Table 2).

Table 2. Characterization vectors of the pilots

Customization factor	Characteristic	Pilot 1	Pilot 2
Domain characteristics	Application type	Information system	Computation intensive system


	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 10 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

Customization factor	Characteristic	Pilot 1	Pilot 2
	Wireless services domains	End user services	End user services
	Business area	Mobile online trading services	Mobile online entertainment services
Development characteristics	Project type	Client - New Server - Adaptation	Client – Adaptation Server - Adaptation
	Architecture	2-tier	N-tier
	Application elements	Java midlets	Java midlets, EJB
	Requirements technique	Structured text / UML cases, Intended screens	Structured text / UML cases
	Wireless networks	GSM, GPRS, UMTS	GSM, GPRS, UMTS, Wireless Lans
	Wireless protocols	TCP/IP	TCP/IP
	Mobile devices	Smart Phones, Cellular Phones	PDA, Smart Phones, Cellular Phones.
	Testing environment	Emulators, real devices	Emulators, real devices
	Implementation language	J2ME, TALAPI (TAL Application Programming Interface)	J2ME, J2EE
	Validation technique	Black box testing, white box unit testing, feature testing on target terminal	Black box testing, white box unit testing, feature testing on target terminal
Enterprise characteristics	Organizational context	Investnet-Italy	Motorola GSG-Italy, VTT Electronics - Finland
	Business objectives	Capturing knowledge, producing high quality products	Capturing knowledge, reducing time to market, producing high quality products
	Role	Service provider, application provider, content provider	Service provider, technology provider
	Experience in software development	5<X<12 years	> 2 years

A characterization vector contains customization factors. These customization factors have been grouped into categories in order to understand their nature better. The definition of the categories and the customization factors for the WISE project are presented in chapter 7. The validation, refinement and extension of the customization factors and categories will be possible as more experience from the development of the pilots is gained.

The characterization vectors are outputs of the activity *set-up pilots*. The benefits from the characterization vectors are:

- The characterization vectors can be used for searching evidences, tools, techniques, and methods from projects with similar contexts.
- The characterization vectors define the scope of validity for experience gained in the pilots.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 11 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- The characterization vectors help process engineers to prepare and understand the scope for designing processes.

1.6 STRUCTURE OF THE DELIVERABLE

The rest of the deliverable is structured as follows:

Chapter 2 describes the execution and new results of IESE's method activity *search and evaluate processes and practices from related fields* from the third iteration, used to perform the WP activity called "Survey on Processes for Software and System Engineering". This activity is intended to find existing evidence of work related to the WISE domain.


Chapter 3 presents IESE's results regarding the empirical approach for creating a new reference process model to develop wireless Internet services. The activities performed as part of IESE's method are, *elicit existing process knowledge, analyze commonalities and differences, and create reference process model*.

Chapter 4 explains the relationship between the process model and the measurement program. It describes how the indicators and metrics were defined in the process model.

Chapter 5 presents the evaluation of quality related activities, techniques (testing, inspections, reviews, prototyping), and indicators found for every pilot during the second iteration.

Chapter 6 presents the conclusions and future work. Appendix one presents the characterization vector definition.

Keywords: software, process model, reference process, process modeling, WISE, Wireless Internet Service Engineering

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 12 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

2. SURVEY PROCESSES FOR SOFTWARE AND SYSTEM ENGINEERING

In order to collect experiences that contribute on the creation of a model for Wireless Internet Services, it is important to consider internal input received from the pilot projects while developing its pilots, as well as external input provided by similar contexts or projects such as: methodologies, tools, techniques, or practices. In other words, it is not enough to use the experiences and practices from the pilot's developers, but also the existent body of knowledge on the domain. This combination of previous and actual experiences allows the software organizations that perform the processes not only to follow their beliefs but also to learn and compare their experiences with previous ones. This section covers the search of relevant literature, which can contribute to the task of designing a software development process for Wireless Internet Services.

2.1 DEFINITION OF THE LITERATURE SEARCH SCOPE

At the beginning of the literature search, the scope was defined as "Software/System Engineering Processes" for "Wireless Internet Services". Due to the recentness of the domain, not many results were found; therefore, a change of the scope was needed. Thus, "Software/System Engineering Processes" for "Wireless" or "Internet Services" were searched separately. The number of results was estimated to be high; therefore other search criteria were introduced:

- A flexible process in order to produce products on time to market
- A process that helps producing products of high quality


The lack of testing techniques, tools, processes, experienced by the pilot partners, during the development of the pilots in the first two iterations, and the importance of this issue, demanded to refocus the scope of the survey in this direction. Therefore, an extensive survey on techniques for testing wireless Internet services [77] was performed at the end of the second iteration, with the purpose to be used by pilots during testing in the third iteration.

2.2 DEFINITION OF INFORMATION SOURCES

The literature search scope was oriented towards technologies that, regardless of the complexity of the environment, are means for producing wireless Internet services of high quality and on time. The criteria to select the articles and include them in the results were: 1) The article presents an approach viable to be integrated in the projects. 2) The article gives a clue for creating a flexible process model that creates products of high quality, on time. 3) The article explicitly describes techniques, tools, guidelines, steps, procedures or methods. A combination of different methods was used in the literature search in order to find as much information as possible. It is important to note that there is no perfect method for finding all the information related to a specific subject [17]. Keyword searches were performed in the following databases: INSPEC, TEMA, COMPENDEX, COMPUSCIENCE, and NTIS. A manual search was performed on the editions between 1999 and 2003 of the following journals: IEEE Internet Computing, IEEE Transactions on Software Engineering, IEEE Software, IEEE Internet Mini Conference, IEEE Distributed Computing Systems, IEEE Computers and Communications, IEEE Wearable Computing, IEEE Technology and TeamWORK conference, IEEE Mobile computing Systems and applications. Other sources of information were international conference publications, software engineering related journals, software engineering consultancy firm reports, software engineering institute reports, and wireless Internet services industrial key player reports. Table 2 presents an overview of the articles found.

Table2. Articles found

Source	Number
International Conference Proceedings	4
Proceedings of ACM	7
IEEE Internet Mini Conference	1

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 13 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

IEEE Distributed Computing Systems	1
IEEE Computers and Communications	1
IEEE Wearable Computing	1
IEEE Technology and TeamWORK conference	1
IEEE Mobile computing Systems and applications	1
IEEE Internet Computing	11
IEEE Software	9
Computers Communications	3
IEEE Transactions on Software Engineering	2
Information and Software Technology	2
Other (industrial key players, consultancy firms)	5

At the moment, from the point of view of process models, this sample is considered to be representative although it is not too big. It is possible that other papers of the same scope were not included in the results. Possible reasons for this are:

- Description of process models that are not published by unknown researches.
- Description of process models that are only of private use inside an organization. These process models are considered an organizational strength, therefore they cannot be revealed.
- Description of process models that were a failure, and therefore were not published.

For any software development organization the benefit of the literature search thus becomes obvious before the start of a new project and during the development.

The objective is for a software organization to be able to identify those results from the literature search that best fits the gaps in their process. These gaps are, e.g., lack of techniques, tools, and procedures, among others. Then, if the result fits into a similar characterization context, it could be integrated into the process model.

2.3 ANALYSIS AND CLASSIFICATION OF THE RESULTS


Below, technologies are described for the following process levels and phases: Life cycle processes, requirements, design, implementation, test, and managerial processes. For each subsection, a description of the characteristics of wireless Internet services is given, and a set of suitable practices found in literature is presented. The characteristics of the wireless Internet services domain are relevant, for example, for choosing the right life cycle process model to follow, or the most appropriated technique for doing requirements, design, etc. As mentioned before, the results of the literature search must be used for providing guidance on suitable practices, which can help software organizations to develop wireless Internet services.

2.3.1 Life Cycle Processes

2.3.1.1 Wireless Internet Services Characteristics

2.3.1.1.1 Flexible Processes

The market of wireless Internet services is expected to grow in accordance with technology, and to be limited by time constraints of the market. Software development organizations must have the possibility to change their strategy in a given circumstance, in a given point of time during development. Strict, unflexible process models like the waterfall model are not suitable for such a context. Organizations must react to the context in the most appropriate manner, and that is only possible through flexible processes.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 14 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

The market of wireless Internet services is expected to grow in accordance with technology, and to be limited by time constraints of the market. Software development organizations must have the possibility to change their strategy in a given circumstance, in a given point of time during development. Strict, inflexible process models like the waterfall model are not suitable for such a context. Organizations must react to the context in the most appropriate manner, and that is only possible through flexible processes.

2.3.1.2 Practices

Suitable life cycle processes for the wireless Internet services domain are the throwaway prototype model and the incremental development model [38], which were found suitable for domains of similar characteristics like Internet and mobile phone [6], [10], [26], [19], [3], [31]. Through these models, essential operational functions are provided initially, and then more capable versions of the system. Increments are usually defined as an agreement between the customer and the development organization. This allows development organizations to get feedback from the final customer during the development of the increments until the final version of the solution is delivered. Additionally, monitoring and controlling the project plan can be done more precisely, and the quality of increments can be assured with the established verification and validation activities.

Agile development practices and techniques aim at finding a balance between flexibility and structure in the actual business environment, where volatility and uncertainty increase [55].

One of them, adaptive software development (ASD), 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. The dynamic systems development method (DSDM) [56] contains three major phases: functional model iteration, design and build iteration, and implementation. All three of them are iterative. DSDM is similar to ASD because it considers collaboration, learning, and allows to introduce new functionality (new requirements) in the project as new things are learned. Additionally, prototypes built for each feature are preferred over long documents as documentation.

Another agile approach, *Extreme programming*, is proposed by [35] as suitable for Web-based projects where time to market plays an important role. *Extreme programming* focuses on producing source code and test drivers, avoiding documentation, and handling the volatility of requirements through small releases. *Extreme programming* reflects an incremental development model. Development cycles are short and based on requirements that are supposed to generate business value for the customer. One risk of agile approaches is that they rely on the tacit knowledge of developers [37]. This should be carefully considered, especially because developers are still learning due to the immaturity of the wireless Internet services domain. Additionally, issues like scalability and performance have to be carefully designed.


The spiral model [36] assumes risks as the driving force of software projects. This model proposes ongoing refinement of the system specification into source code components. Refinements are made through cycles, and each cycle is risk assessed. A risk assessment determines if a project continues or is cancelled. The nature of the spiral model seems reasonable to apply in a convulsionate domain like wireless, but the real cost of identifying, analyzing and maintaining risks can be high, which is not so suitable for small and medium-sized companies.

Boehm [37] proposes a combination of agile and plan-driven methods through risk-driven spiral methods, which are intended to balance flexibility and discipline. The rationale behind that is that although the market changes constantly and puts pressure on development organizations to deliver their products rapidly, increasing dependability of systems and applications demands high quality products.

2.3.2 Requirements

2.3.2.1 Wireless Internet Services Characteristics

2.3.2.1.1 High Volatility of Requirements

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 15 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

Wireless Internet services will be deployed across heterogeneous networks, heterogeneous clients, and for all types of users. The business cases are not clear yet. The stakeholders (i.e., network operators, service providers, application providers, technology owners) are still researching possible business cases that promise revenues in this new growing market. All this uncertainty is reflected in high volatility of the requirements. This phenomenon was also seen in the Internet domain. A flexible and agile mechanism to specify requirements early or late in development is needed.

2.3.2.1.2 Usability

Small screens and limited keyboards are important constraints that are inherent to mobile devices. On the Internet side, many applications have problems such as improper user interaction mechanisms, other web sites are visited only once by users because they do not find quickly what they are looking for, or the interface cannot be displayed in the browser. Additionally, fluctuations of the Wireless Network could enhance or degrade the performance of the service. Therefore, Wireless Internet Domain developers will have to invent practical ways of user-software interaction, data exchange, and web site navigation. The requirements specification process must consider these aspects.

2.3.2.1.3 Device Independence

An application is device independent if it can be deployed on any mobile device, and its functionality and capabilities are kept. Web content and application authors face a similar problem when trying to display an entire system on different browsers. Many pages have to be reworked. Wireless Internet Services face a more complex problem because of the diversity of devices on the market. It is unaffordable for software development organizations to rework their software for every new device.

2.3.2.2 Practices


Index cards are proposed by the usage-centered design approach [22], as a mechanism to specify requirements as part of an agile usage-centered engineering approach for Web applications. Here customers, managers, and developers collect the requirements on cards during a brainstorming session, where they sketch the application's purpose from a business point of view, and express their wishes regarding functionality, features, content, and capabilities. The cards are sorted and clustered. The clusters are taken as the basis for specifying user requirements functionality.

Extreme programming [21] proposes user stories as a medium to capture functional requirements in a simple, non-formal language. The developer writes them with the collaboration of the customer. The user stories are written on index cards where the tasks of the system are described. These stories are the basis for planning iterations, tracking progress, specifying and testing the functionality. User stories seem to be suitable for requirements that come late in the development of the application [27]. The usual question is, how scaleable will the system be afterwards?

The index cards and user stories involve customers under the assumption that they are participative and proactive and that they actually represent the user's needs. This might not be true in all of the cases. The review of requirements by experts could compensate this problem.

In order to address usability, [25] presents the user-centered approach, where the business development group of a software organization is in charge of studying and defining the profiles of its possible users. The profiles are used to determine possible tasks and goals of the users, specifying the functional requirements, and creating a prototype for user analysis. The prototype consists of user interfaces that will be discussed with the users and then implemented according to the feedback. In a heterogeneous market like wireless Internet services, this approach could be of great help, because it forces development organizations to consider a wider spectrum of possible profiles.

Combinations of the previously mentioned approaches could be applied depending on the context of the software development organization. For instance, a small software development organization has fewer resources for looking at the requirements using the market division than a large organization. Certainly organizations that want to enter the wireless Internet services domain must know that one real concern and possible factor of success is the usability of the application.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 16 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

Regarding device independence, a good starting point for clarifying the concepts is given by [24]. It is a survey that presents a classification of available technologies, and their relationship with device independency in the context of wireless Internet services. For example, device attributes like output, input, processor, memory, multimedia objects, application language, or browser language influence the degree of independency of an application.

Devices receive content as multimedia objects, application languages or browser languages. Depending on the underlying hardware, devices are able to use different types of content. Therefore, in order to achieve device independency, the content must be sent in a compatible format for a given device.

There are technologies that can be used to adapt the content or application according to the device capabilities. Content adaptation can be done in the server, proxy or client browser. Some examples of these technologies are: HTTP request header files, CC/PP composite capability preferences profile, WAP UAPROF, SyncML, and Universal plug and play.

The mentioned survey is a good reference for understanding how each of the mentioned technologies can help when trying to deliver a device independent application. The bad news is that at the moment, there is no dominant/unique standard, therefore, choosing a specific technology can imply high risks.

In order to mitigate this risk, the W3C consortium is working on an initiative focusing on device independency and standardization. The idea is that web content and applications are accessible anyhow and anytime. Anytime refers to many access mechanisms (i.e., heterogeneous clients that can provide access anytime), and anyhow refers to many modes of use (i.e., audio, voice, touch, among others). One product of this effort is the Device Independency Principles document [32]. 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.

Use cases are proposed by [6], for describing the functionality of the application. This is a more traditional approach where, after some interviews with the customer, the developer describes the functionality of the application. No recommendations are given in order to address usability, device independence, or high volatility of requirements, even though the model proposes accepting requirements late in development.

2.3.3 Design

2.3.3.1 Wireless Internet Services Characteristics

2.3.3.1.1 Scalability

Usually, if an Internet site is successful, high consumption of primary systems resources (CPU, memory, file system bandwidth, and network bandwidth) is expected. Wireless Internet services will run on top of Internet sites. The same behavior is thus expected for successful wireless Internet services. Therefore, wireless Internet services sites should be designed to be scalable.


2.3.3.1.2 Seamless Mobile Services

Aiming at seamless/transparent mobile services requires mechanisms to hide heterogeneous and changing contexts.

Wireless Internet services can be provided on weakly interconnected low-speed networks such as GSM or high-speed networks such as UMTS [27]. As the user moves, changes from a faster network to a slower network are obvious. Mobility of the terminals has consequences on the product model (a component to follow and track is needed in the architecture). Thus, while designing a service for mobile devices, which requires data exchange over the network, the current location of the user, and the bandwidth available on the wireless network, are two issues to be considered, among others.

2.3.3.1.3 Charging and Billing Models

The evolution of mobile networks and the transformation of mobile devices into Internet terminals have created the need for a new billing and charging model, where the subscribers are charged not only for the time and quality of the telephone usage, but also for the Internet services they use (i.e., email, Internet

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 17 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

browsing, multimedia messages). The need for very well defined components, interfaces and communication mechanisms between components is important in order to create charging and billing mechanisms capable of supporting actual or future business models.

2.3.3.1.4 Usability

As already mentioned in the requirements phase, physical attributes of mobile devices, especially, are a challenge for creating usable applications, but the structural design of a wireless Internet service site, which determines how users will navigate within it, is also an important activity. The system should be designed to allow the users to find what they are looking for quickly. The complexity of this task depends on the size of the site and its relationships with other sites.

2.3.3.1.5 Device Independence

A discussion was already introduced on this subject in the requirements section. Findings oriented towards designing for device independence are presented in this section.

2.3.3.2 Practices

A survey performed within 25 organizations in the UK [26] revealed that formalized design web techniques like hierarchy charts, site flow charts, and storyboards were used in the web domain. Hierarchy charts were used to relate web pages of a site. Site flow charts sketched the decisions to reach certain functionality, and storyboards contain the sequence of web pages that a user will encounter within a web site. These techniques were used basically to design the navigation of the structure.

Some of the studied companies had developed website layout standards for using video, animation, graphics, colors, and navigational standards such as where to place the back button, and the use of banners and menus. Standards for designing web site content were found, as, for example, the use of specific keywords.


One of the major conclusions of the study is that few organizations have guidelines or standards for web site development and ad hoc practices are dominant. A similar scenario can be expected in the wireless Internet services domain. Ad hoc approaches can lead the development of complex and unmaintainable sites. Structured techniques/guidelines for design are necessary sooner or later.

Examples of the use of structured techniques are given by [6], [27], [8], and [2]. They have in common the use of object-oriented principles to design static and dynamic views of a wireless Internet service application. Patterns like the MODEL-VIEW-CONTROLLER are recommended for use in wireless Internet service applications by [27], 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. Other techniques use patterns to customize the role of a user, and the structure, behavior and links of a page [8].

Regarding device independence, Giannetti [13] provides the 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.

ScalableWeb is a technique presented by [5] 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 [39].

Mori et al. [15] present an *XML-based approach* oriented to design applications that are device independent. The tool TERESA (Transformation Environment for Interactive Systems Representations) provides a semiautomatic environment supporting the presented method and transformations.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 18 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

One commonality between *ScalableWeb* and the *XML-Based approach* is the specification of a task model as input for creating the device presentation model, or the abstract user interface, that are later transformed into the device specifics with the help of an automatic/semiautomatic tool.

All of the approaches agree on the need of a high level description of content, style and interaction that allows adaptation.

At the moment, topics of research in the device independency area are, for example, how to balance independency with the usability of the application. One application may appear as expected in the devices, but the usability experience might not be as satisfying for all of them. What are the steps to create abstract, general presentation models? Should the generation of the specific presentation models be totally automatic, or just partially, so the authors could manipulate it?


Regarding scalability, [4] introduces what they call "Scalability design process" 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 paper contributes with a set of guidelines for system partitioning, i.e., dividing the system into components with a well-defined interface and functionality. 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.

In order to build seamless mobile services, Friday et al. [2] propose techniques 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 bandwidth situation. The adaptation techniques were validated through the development of a mobile collaborative system. One of the final conclusions of the study was that mobile systems must have the support of adaptation techniques at all levels, in order to be effective, but architecture to propagate QoS information through the system is still required.

A discussion of billing infrastructure and charging models for the actual and future Internet, and how they could be modified for being used in wireless Internet services, is presented in [33]. One interesting example is the Paris-Metro charging model proposed by . 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 [33] 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.

The usage-centered engineering [22] approach presented in the requirements phase that addresses user interface usability does continue in design. Designers must produce a role model, a task model, and an abstract model. The role model groups the common characteristics of user interaction with the system in roles. These characteristics are related to the purpose, duration, attitude toward the system, and information exchange between the user and the system. A task model is a set of task cases and their relationships. A task case lists the steps of the system to provide the desired functionality without assumptions about the user interface. Finally, the abstract model describes the user interface with interaction contexts. The abstract model does not contain details about the look and behavior of the user interface. Designers use these models to create a comprehensive user interface.

Nerurkar [30] suggests merging the GUI methodologies used for designing traditional systems with the new Web design techniques [41], 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.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 19 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

More specific guidelines for designing user interfaces can be found 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 [45].

The previous approaches gave guidelines for producing usable sites, but how could that be measured? The *Card Sorts* technique is proposed by [23] 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.

2.3.4 Implementation

2.3.4.1 Wireless Internet Services Characteristics

2.3.4.1.1 Programming languages and protocols

There is a diversity of platforms, programming languages, and protocols already available for developing mobile applications and Internet services [27], [24]. Software developers have to answer questions like: Is WML suitable for implementing a wireless Internet game? If yes, which are the preconditions, or technical requirements? If not, what is the best suitable technology?

2.3.4.1.2 Technical constraints: Power, memory storage, and security

It is not enough to have a good design in order to optimize resources. It is also important to look at programming techniques, programming standards, or tools for tuning and optimizing the produced code.

2.3.4.1.3 Flexible communication channels between developers

Lack of knowledge and experience must be addressed in software development organizations with communications mechanisms that allow developers to resolve problems quickly and efficiently. The experience of developers or the will to work collaboratively is not enough.

2.3.4.2 Practices


There are plenty of Internet services for providing financial, weather, or sport information to clients, i.e., services that require little user interaction. These services 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 [43]. WAP 1.x uses the Wireless Markup Language (WML) for document formatting. 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.

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 browsers, but not all HTML features can be converted into XHTML.

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.

Wireless Internet services that demand interaction of the users, like games, need a more flexible programming platform. 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 client should be deployed for every possible platform, slowing down time-to-market of the service and increasing costs. JAVA, on the other hand, is commonly used because of its

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 20 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

portability. Especially after the release of J2ME, Java can now be deployed on many wireless devices providing a common ground for developers.

Regarding device resources, although J2ME has been optimized, it still demands considerable capacity from the processor. In order to optimize the use of the device power, tips, guidelines and techniques can be found in the J2ME/WAP developer discussion groups. One example can be found in [44]. Memory storage is another constraint that tends to improve with technology evolution. Techniques used to reduce 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 that can be useful [50].

Security for WAP applications can be assured through the Wireless Transport Layer Security (WTLS). In the case of J2ME applications, power and memory constraints make security a challenge.

Pair programming can be seen as difficult to implement in an industrial context, but in a new domain such as the wireless Internet, it could perhaps have a good effect in order to resolve new and uncertain problems thanks to its person-to-person communication mechanism. Pair programming is an extreme programming technique where two developers produce code in one machine [35]. 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.

Zettel et al. [19] propose the *LIPE* process model to develop e-commerce services on time to market, based on other extreme programming techniques like *refactoring*, *realize scenario*, and *rework code*.

Although extreme programming techniques seem to be flexible enough, there are some drawbacks. In the case of *refactoring*, small teams formed by great developers can be successful [36]. But what if the developers are not so experienced? It has been concluded by [41] that agile practices are not intended for larger teams. Although success cases for larger teams have been presented [42], this assumption must be considered especially in the wireless Internet Services Domain, where projects can become large and complex.

2.3.5 Test


2.3.5.1 Wireless Internet Services Characteristics

2.3.5.1.1 Testing in a realistic environment

A considerable amount of effort is required in order to prepare realistic conditions for a wireless Internet service to be tested. Consider a service provider that wants to test its new weather information service. The service should run on laptops, mobile phones and PDAs. Additionally, the service provides wheather location related information. This scenario already implies important assumptions to be made while testing the application. Different devices, different wireless networks, and different user profiles. There are also many context factors that influence on the results of the test, e.g., network traffic, user location, and mobility. Recreating scenarios with typical situations requires effort and money for setting up the appropriate hardware and software.

2.3.5.1.2 Usability

Usability can be defined as the degree to which a given piece of software assists the person sitting at the keyboard or having a wireless device such as PDA or mobile phone to accomplish a task, as opposed to becoming an additional impediment to such accomplishment. To date mobile portals have been characterized by their poor usability and the limited online experience offered to end-users. Some causes have been responsible - unreliable early handsets, limited content, slow connections, and poor portal navigation. Today, the first three of these causes have been somehow addressed (by improved handsets, better content, and high-speed infrastructure) but portal navigation remains a problem, with users routinely expected to make more 'clicks' on their mobile handset in order to locate content, thus greatly limiting their ability to easily locate, and benefit from, wireless content. Therefore, previous testing to deliver must be oriented to friendly end users, in order to discover possible rejection criteria.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 21 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

2.3.5.1.3 Energy Consumption

Mobile devices have limited processing power and memory as well as energy consumption constraints. Energy constraints are an important issue in software design for mobile devices. It is essential to design software to minimize energy consumption, preventing hardware accesses from depleting the battery of the mobile device faster than necessary. For this reason, extraneous hardware accesses that could be ignored in a desktop environment have to be tested much more closely in a mobile environment.

2.3.5.2 Practice

Finding mechanisms or tools for simulation or emulation of scenarios can help to solve the problem and reduce costs for setting up a realistic testing environment. Technology providers provide emulators of wireless networks as well as of many mobile devices on the market.

Emulators approximate the functionality of the real device, or networks. In practice, using network emulators can be of great help for creating tests, collecting the results, and repeating them under different network conditions.


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. The emulator applies 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 can be used as central-control wireless. General-purpose network emulators falling into this category are ONE [81] and Dummynet [82]. Typical network parameters supported by these central control wireless emulators include packet delay distribution, packet drop pattern, bandwidth, and queues.

VINT/NS [83] is one of the most commonly used simulation combined wireless emulator. 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 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. This approach lacks the support for the evaluation of real topology-related protocols.

Trace based mobile network emulator [84] also emulates the characteristics such as performance and 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. The first phase collects traces from a target mobile wireless network. Trace collection logs every outgoing and incoming packet, along with the time at which it was sent or received. Other protocol related information such as sequence numbers, or flags are also collected. The second phase constructs a wireless network model with the collected traces. With this network model it must be possible to obtain the parameters solely from observations at endpoint. It should be cheap to compute the parameters, and to use them during modulation. The last phase reproduces the traced network effect in a wired network.

Another emulator named flying emulator [85] can be used 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. Moreover, the framework allows emulators to easily simulate other characteristics of wireless networks by using a runtime bytecode rewriting technique.

One distributed network emulator system, EMPOWER [46], 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 [44]. EMPOWER allows the user to define packet latency and bandwidth as parameters and test a given topology wireless network. There are plans to include a physical layer simulator in order to improve the emulator capabilities of wireless networks.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 22 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

A description of the Model Based Testing technique, and its application for testing a Pocket PC application can be found in [47]. 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.

Not all applications and protocols can be tested with emulation as mentioned above. Simulation can be an alternative approach. There are two important approaches to simulate the network behaviour for application testing: complete network simulation including the radio frequency [86] and network behaviour simulation on IP level [87]. The first approach 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. In this way end-to-end push or pull application can be triggered with the simulator acting as the mobile network bearer. The second approach captures the IP packets between the client and the server and introduces the network related interruptions or error scenarios.


Simulation can also be done to test characteristics like usability. One visualization-based approach to improve the usability of a web site, and a predictive model to locate problem areas, is introduced by [9]. The approach underlines the importance of using visualization techniques to understand the behaviour of users on a web site and to identify unreachable places. Visualization techniques can also be used to analyze the past behaviour of a site, and to understand the impact of new changes.

Concerning how to test the usability of wireless Internet services, the web usability assessment model [78] 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.

The usability of mobile devices as a medium to deploy WAP applications has been criticized because of their physical limitations by [52], and [53]. Their usability test and studies reveal that bigger and more capable user displays improve user interfaces, and therefore acceptance by users. An interesting study done by [51] was focused on detecting usability problems through testing, and afterwards improving the application. In this study, the users tested alternatives of a user interface with the same functionality. Users were gathered, and performed the same activities within a wireless Internet service. Meanwhile the time and number of interactions were measured. Finally, users were interviewed about their experiences and suggestions. The measures and information were taken as basis for deciding which was the most suitable interface and for creating guidelines to be used in future projects.

Markov models [79] are mathematical models, which can be used for the validation of the usability of the mobile devices. 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 [80].

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 [88]. Applications interact not only with the display, but also with various other hardware components: processor, memory, network interface, and possibly hard drive. All these components consume energy during operation. This gives an opportunity to monitor and record power levels during the test execution. The recorded power levels can then be used to validate energy requirements. Energy quality requirements specify that the software should execute with minimal power levels and energy consumption.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 23 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

2.3.6 Managerial Processes

2.3.6.1 Wireless Internet Services Characteristics

2.3.6.1.1 Small increment planning

As already mentioned, the incremental development model is suitable for domains where high volatility of requirements, lack of experience and time to market are success factors. What is the best increment to be delivered? What is a realistic time interval for each increment? How to select a consistent set of requirements for the increment? These are questions that have already been addressed in the area of requirements engineering and applied in areas like Internet or Mobile. Regarding wireless Internet services, a software development organization can have more than one role, for example as application provider, service provider, and integrator of services. This characteristic demands vision and discipline for releasing the product, and impacts the procedure to determine the best suitable or most profitable increment.

2.3.6.2 Practices

One planning method for increments called *Construction Planning* is presented by [11]. 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.

A tool for understanding the nature of planning releases was developed by [48], and tested in three case studies. Based on the assumption that cost and value were the most important factors for deciding what is a release, developers were allowed to assign cost and value to each requirement, and to prioritize them. The tool would find a set of possible releases, which were reviewed and validated by the developers. One important statement is the definition of the release planning activity as a *wicked* problem [49]. 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 unrepeatable, therefore no measures of success are possible. This was confirmed in the case studies. It was seen that the number of variables influencing the definition of a suitable release could be very large, and that every release was influenced by new variables, or the context changed the values of the existing ones. Software organizations could benefit from this area of research as it addresses one key activity for the development of wireless Internet services.

2.4 SUMMARY OF PRACTICES

Table 3 presents the results of the literature search. A result should be read in the table as follows: Source **Hammar** [25] taken from the **Internet Services** domain contributes with a **Process**, to be used during **Requirements, Design, Implementation**, and enhances **time to market, quality and usability**.

A list of possible values for each column is given in the following:

Domain: These are the related domains where the literature was found, like mobile, agile, telematic, wireless, Internet, and Wireless Internet.

Contribution: For example, guidelines, a technique, methods, processes, models, roles, and tools.

Phase: These are the spots in the process where the contribution are presumed to be helpful for the development of wireless Internet services

Possible success factors: Based on justifications found in literature [20], [2], [16] and on the objectives of the observed projects, four possible success factors were used: Time-to-market, interface usability, device independence, and quality.

The results presented here, are though to be used by software development project managers, in order to



	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 24 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

TABLE 3
LITERATURE SEARCH RESULTS

Source	taken from the domain	contributes with a	to be used during					and enhances
			R	D	I	T	M	
Hammar [25]	Internet Services	Process	x	x	x			Time to market, usability, quality
Constantine et.al. [22]	Internet Services	Process	x	x	x			Usability, quality
Beck [21]	Agile processes	Process	x		x	x		Time to market
Cockburn [1]	Agile processes	Process	x		x	x		Time to market
Adamopoulos et.al. [6]	Telematic Services	Process	x	x	x	x		Quality
Taylor et.al. [26]	Internet Services	Processes		x				Time to market
Kovari et.al. [27]	Wireless Internet Services	Techniques		x	x			Usability, quality
Schwabe et.al. [8]	Internet Services	Process		x				Time to market, usability, quality
Friday et.al [2]	Mobile Services	Techniques		x				Usability, quality
Cushnie et.al. [33]	Wireless Internet Services	Desing Model		x				Usability, quality
Maurer et.al. [35]	Internet Services	Process	x		x	x		Time to market
Nerurkar [30]	Internet Services	Process		x				Usability, quality
Rosenfeld et.al. [41]	Internet Services	Techniques		x				Usability, quality
Sun Microsystems	Wireless Internet Services	Guidelines		x				Usability, quality
Upchurch et.al. [23]	Internet Services	Process		x		x		Usability
Buttler [24]	Wireless Internet Services	Technologies		x	x			Device independence, time to market
W3C [32]	Internet Services	Guidelines		x				Device independence, time to market
Gianneti [13]	Internet Services	Process		x				Device independence, time to market
Wong et.al. [5]	Internet Services	Technique		x				Device independence, time to market
Wong et.al. [39]	Internet Services	Tool			x	x		Device independence, time to market
Mori et.al. [15]	Internet Services	Technique		x	x			Device independence, time to market
Roe et.al. [4]	Internet Services	Technique		x				Quality, time to market
Zettel et.al. [19]	Internet Services	Process	x		x	x	x	Time to market, usability, quality
Sun Microsystems [44]	Wireless Internet Services	Guidelines			x			Quality
Zheng et.al. [46]	Wireless Internet Services	Tool				x		Quality
El-Far et.al. [47]	Wireless Internet Services	Process				x		Quality
Chi [9]	Wireless Internet Services	Tool				x		Quality

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 25 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

<i>Karlsson [11]</i>	<i>Mobile Services</i>	<i>Process</i>						x	<i>Time to market, quality</i>
<i>Carlshamare [48]</i>	<i>Market driven domains</i>	<i>Process</i>						x	<i>Time to market, quality</i>
<i>Becker [78]</i>	<i>Wireless Internet Services</i>	<i>Tool</i>						x	<i>Usability, quality</i>
<i>Thimbleby [79]</i>	<i>Wireless Internet Services</i>	<i>Technique</i>						x	<i>Usability, quality</i>
<i>Kohler [81]</i>	<i>Wireless Internet Services</i>	<i>Tool</i>						x	<i>Quality</i>
<i>Rizzo [82]</i>	<i>Wireless Internet Services</i>	<i>Tool</i>						x	<i>Quality</i>
<i>Fall [83]</i>	<i>Telematic Services</i>	<i>Tool</i>						x	<i>Quality</i>
<i>Noble [84]</i>	<i>Telematic Services</i>	<i>Technique</i>						x	<i>Quality</i>
<i>Satoh [85]</i>	<i>Wireless Internet Services</i>	<i>Tool</i>						x	<i>Quality</i>
<i>Puuskari [86]</i>	<i>Wireless Internet Services</i>	<i>Technique</i>						x	<i>Quality</i>
<i>Tyago [87]</i>	<i>Wireless Internet Services</i>	<i>Technique</i>						x	<i>Quality</i>
<i>Sinha [88]</i>	<i>Wireless Internet Services</i>	<i>Technique</i>						x	<i>Usability, quality</i>

R = Requirements, D = Design, I = Implementation, T = Testing, M = Management


3. DEFINE NEW PROCESS TO DEVELOP WIRELESS SERVICES

The objective of the activity *set-up pilots*, as described in section 1.5 is to design suitable projects for observing software developers in action, what techniques or processes do they follow in order to accomplish their objectives. This observation must be done systematically, first by eliciting the process models, then by comparing them through an analysis of commonalities and differences, and finally by creating the reference process model. Each one of these approaches is explained in this chapter, and the tangible result, which is the reference process model, can be seen in part B of this deliverable.

3.1 ELICIT EXISTING PROCESS KNOWLEDGE

The identification of existing processes consists of two stages, *orientation* and *detailed elicitation*. During the orientation phase, a process outline is developed. The process outline provides an overview of the process and facilitates further elicitation activities. For example, process information can be described with the help of the process-modeling schema implemented in the Sparmint [28] tool. The outline will help the process engineer to elaborate sample interviews and to select the process performers to work with in the next stage called *detailed elicitation*. If weaknesses in the current process are already known, they should be eliminated. Thus, during the interviews process performers should already be asked which practices in the current process they consider inefficient.

Subsequently, the process model is reviewed: People who provided information for the model are asked to review the model to make sure that all information captured was correctly transformed into the model. The result is a description of pilot processes as they are actually being performed in the respective environment. One further benefit of this method is that involving process performers early increases process awareness among them. Moreover, involving process performers in the definition/tailoring of the process can lead them to more strictly follow a process that they somehow have helped to define, rather than a process that was externally defined and imposed.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 26 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

For each pilot project, a description of its process model as performed for the second iteration was elicited. The main information sources used were interviews with process performers and managers, and the analysis of documents used or produced in previous projects. In particular, the following models were elicited:

Pilot 1 – online trading – is performed within one organization. Based on an example of this process, an initial process model for Pilot 1 was described.

Pilot 2 – online entertainment – is performed jointly by two organizations. This application consists of a client and a server part, where one organization is responsible for the client, and the other one for the server part. Thus, individual process models were developed for each of the Pilot 2 partners. Figure 3-1 shows the development of the process models for the second iteration.

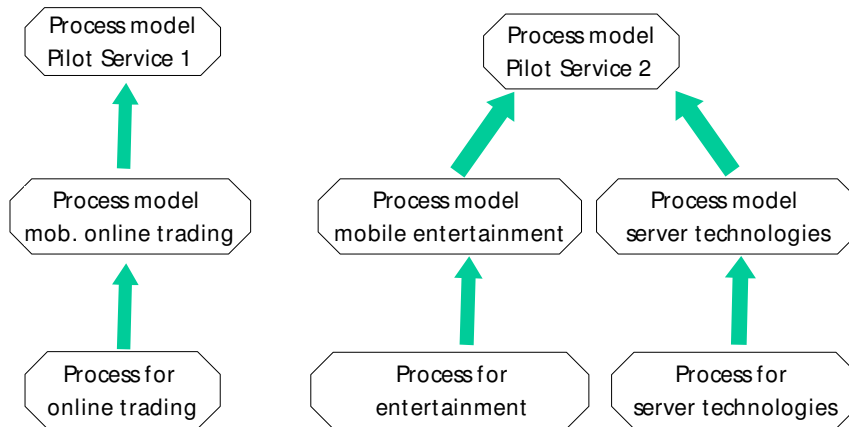


Figure 3-1: Second iteration of process model


3.1.1 Process Models

The process models distributed for the second iteration appear in the part C of this deliverable. To see more details, please refer to this part.

3.1.1.1 Process Model Objectives

In particular, the versions of the process models were developed in order to accomplish the following objectives:

- To obtain a process outline definition
- To describe the process phases
- To describe the main activities and artifacts of the processes
- To describe the product flow between activities and artifacts
- To describe the roles and tools used
- To integrate a first set of measures from deliverable D8 after they have been identified. These measures are:
 - o Calendar time and effort: They are intended for activities.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 27 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- Size: It is intended for artifacts, e.g., number of implemented functions.
- Defects: Intended for artifacts.

3.2 ANALYZE MODELS

The different process models are integrated to form a reference model. To develop a reference process model, commonalities and differences among and between the different process models have to be analyzed. Commonalities between the pilot processes may indicate typical process steps and can lead to abstractions of the pilot process models. Variations of the pilot processes may lead to specializations of the comprehensive process model. In this case, different context characteristics (such as developers' experience, or system type, among others) of the pilots may indicate the reasons for process variations. If processes differ and no context deviations can be identified, the context is probably not characterized completely and there is at least one influence factor on the process that has not been identified yet.

Activities, artifacts, roles, and tools from every pilot are analyzed to obtain valuable information for the reference process model. The process engineer compares descriptive process models, following a rule-based semiautomatic procedure, with the support of a specifically developed new tool, SPEARSIM.

The idea of semiautomatic comparison can be explained as a loop consisting of four steps:






Step 1: A tool should propose which parts of two development process models are similar by providing assumptions. The computation of such assumptions would rest upon rules, which formalize different similarity aspects that may occur between entities of two process models [67].

Step 2: The process engineer should turn some assumptions into facts, i.e., accept or reject some of the assumptions provided by the tool.

Step 3: The facts should be used by the tool in order to perform an improved computation of the residual assumptions.

Step 4: Finally, the tool should present the results of the computation to the process engineer, who has to decide whether to iterate the loop by setting new facts and initiating a new computation or whether to stop the comparison.

The integration of a rule-based comparison approach into the SPEARMINT environment [28] enables the definition of complex rules by means of the conceptual richness offered by the language defined by this environment. SPEARSIM implements the rule-based comparison approach discussed in this section. The tool has become an integral part of SPEARMINT.

Figure 3-2 shows the user interface of the tool. The square icons depicted in the figure represent assumptions as proposed by the system and facts as set by the process engineer. A filled square  indicates great similarity between the two entities on the x and on the y axis, respectively. An empty square  means no commonalities. An icon like  means moderate similarity. Facts are represented by the icons , for different entities, and , for identical entities.

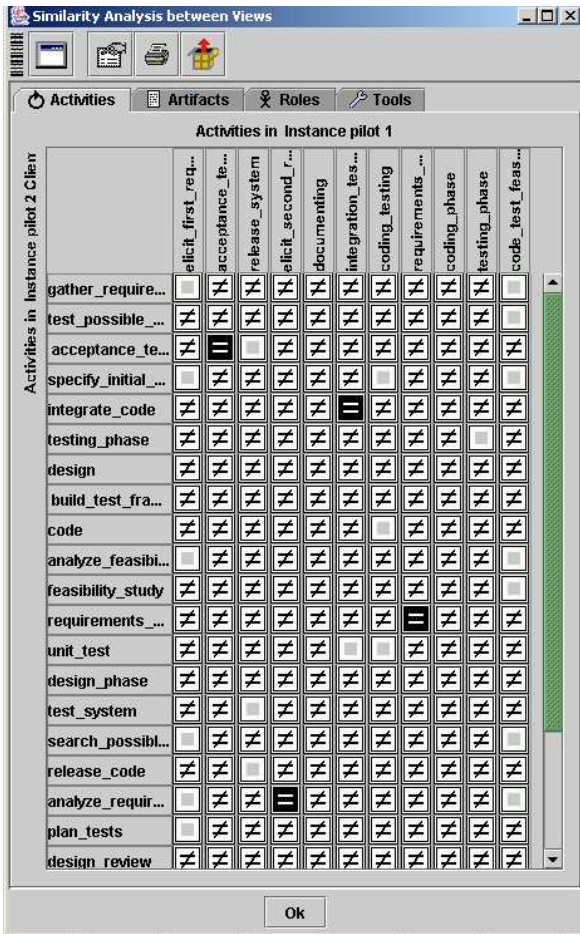



Figure 3-2. SPEARSIM interface

Commonalities can be recognized in the structure of the architecture, which is described in the design document. Both processes have used this design document as input for the coding phase. Logical architecture components for user authentication, billing, accounting, and user profiling can be found in the architectures of both pilots. This has implications on the processes, e.g., the development of respective interfaces has to be considered by the processes. Finally, both pilot processes include an acceptance test in the customer environment.

Other commonalities detected after the third iteration between the two pilot processes are the following: Both pilots have very close involvement by the customers and the providers of technical infrastructure. Market demands need to be carefully examined and understood. Therefore, both pilots introduced a feasibility study. Both pilots used the same structure in order to document the feasibility study. Both pilots agreed on the usefulness of the activity, and underlined it as essential because without it, requirements, estimations, and the architecture are more difficult to specify. These feasibility studies demanded considerable effort for finding a set of possible solutions and coding prototypes in order to test possible solutions. The prototypes were not thrown away; they were reused later during the coding phase. Both pilots underlined the fact that emulators available for testing wireless applications are far from being reliable. Therefore, real devices were used as early as possible in the development process. Pilot 1 migrated from WAP to J2ME and found the same problems of performance and portability experienced by Pilot 2. Workarounds had to be coded and proprietary libraries had to be used to solve these problems. The roles of the development groups from the

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 29 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public


pilots were very similar and had similar responsibilities. The same was seen with the tools used to develop the pilots.

Some of the differences that were encountered were the following: Pilot 2 was a new development oriented to a wider market spectrum, which implies covering more platforms, while Pilot 1 was an adaptation of an existing web service towards the wireless domain. This changed, due to the migration to J2ME; Pilot 1 can now be categorized as a new development. Other process differences have not changed from iteration 1 to iteration 2, as for example: designing, coding and testing, are comprised in one activity for Pilot 1, called coding-testing, without clear entry and exit criteria among each other, whereas for Pilot 2, these criteria are well defined. The same happens with the Pilot 1 activity release system, which is a combination of the Pilot 2 activities release and test system. Different user interface requirements (developing multimedia interfaces requires other procedures than developing pure textual interfaces) and different non-functional requirements (e.g., mobile online trading requires much higher reliability) lead to, for example, different validation tests for the final product: Pilot 1 shall be reliable and guarantee highly secure transactions, Pilot 2 shall guarantee high performance and usability features, therefore different test suites had to be designed to verify that the final products fulfilled specific requirements.

Fig. 3.3 shows the commonalities and differences mentioned above.

Pilot 1	Commonalities	Pilot 2
Roles - Content provider - Service provider	Roles - Service developer Development model - Incremental prototypes Analysis techniques - Careful market analysis - Involvement of customer and infrastructure providers - Feasibility study Implementation language - J2ME Requirements techniques - Structured text, UML use cases Design techniques - Architecture specification UML Testing techniques - Provisional technical infrastructure - Acceptance tests by the customer - White box testing Components User authentication, accounting, billing, user profiling Protocol -GSM/GRS/UMTS Project type - Server adaptation	Roles - Technology provider Business area - Mobile online entertainment services Project type - Client adaptation Implementation language - J2EE Testing techniques - Performance and usability tests - Validation tests Application type - Computation intensive system

Fig. 3.3. Commonalities and differences

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 30 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

3.3 REFERENCE PROCESS MODEL

We use the term software process reference model for a process model that integrates consistent and validated empirical and theoretical evidence of processes, products, roles, and tools used for developing software in a domain. Additionally, such a model should provide tailoring guidelines. Tailoring guidelines describe relationships between variants of the software process reference model and the context of its application in a specific organization. As a prerequisite, an appropriate representation of the reference process model and tailoring mechanisms are required. Identification of specific conditions and effective adaptations can be done based on process designer experience, on continuous feedback from process performers during service implementation, and on available literature and historical data on similar project.


3.3.1 Reference process model creation

The following section presents how the results from the commonality analysis are used in order to integrate the process models and to derive guidelines for future use of the software process reference model. The following activities are proposed:

1. The process engineer selects pairs of similar processes whose value is in the table of commonalities and differences. For each pair, the process engineer asks the process owners whether they believe that the processes have the same goal. If this is the case, a new name and description is written. Next, the process engineer pastes the new common process into the reference process, creating what can be called a common part of the reference process model. If the process owners do not believe that the processes are common, then another value is given to the compared pair of processes.
2. The process engineer selects cases of a process that exists in only one process model, or that is different from all the processes of the compared process model (i.e., a column or a row with only values). For each case, the process engineer asks the process owners whether they believe that the unique process can be added to the reference process model, or whether that is not possible. This question is justified by the fact that the process owner, who does not follow the mentioned process, could see it as an opportunity for improvement. If this is the case, a new name and description is written, and a new common process is included in the reference model. On the other hand, the process owner could have no interest or no resources for following the mentioned process. In this case, the process is added as an optional part in the reference process model. Optional means that the process might or might not be instantiated by process owners.
3. The process engineer selects cases whose value is in the table of commonalities and differences. This occurs with processes whose purpose is similar, but whose procedures are different. They are declared alternative, and included in the reference process model. Alternative processes are those that are similar in purpose but different in procedure.
4. Identify gaps and fill them with best practices found in a literature search: After merging the process models, gaps are expected, especially when the domain is not well understood, and none of the process models contains practices or techniques to solve new challenges.
5. Derive tailoring guidelines: The created model should provide means to identify which project's characteristics force a process owner to enact a given alternative or an optional process. By using the characterization vectors as an attribute of the activities of the reference process model, the process engineer and the process owners identify those characteristics that justify the common, optional, and alternative parts of the reference process model.

4. DEFINE MEASURES AND INDICATORS

The GQM paradigm is a mechanism for supporting the setting of operational goals for software projects. This mechanism helps software organizations to integrate their goals with the process models, products, and quality expectations. The WISE project WP3 contains the creation of GQM plans for pilots 1 and 2, where the indicators to control time, quality, and cost were defined. They support the control of cost, quality, and

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 31 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

time of the process. In order to develop these GQM plans, a definition of the process was needed. Therefore, the process models and interviews with the pilot participants were mandatory inputs. The connection between the measures and indicators and the process model were the data collection sheets. The development of the sheets was also part of WP3. They played an important role for the Process Engineer to monitor the enactment of the initial process models. Inconsistencies detected on data triggered an alarm for the Process Engineer. In this case, the Process Engineer contacted the pilot participants, and after receiving their feedback, updated the process models, and the data collection sheets.

For every pilot, data collection sheets were distributed as follows:

- Excel data collection sheets to be completed by the manager.
- Excel data collection sheet to be completed by every developer.

To get a more detailed look at the indicators, interpretation, and analysis, please see deliverables D8.V2 and D9.V2.

5. EVALUATE QUALITY-RELATED ACTIVITIES

Quality activities considered in this chapter are validation activities and verification activities followed during the third iteration. In the following section, the descriptions of the activities for each pilot are explained in more detail.

5.1 PILOT 1

5.1.1 Verification Activities

There is no evidence of verification activities like reviews or inspections.

5.1.2 Validation Activities

There is evidence that the following activities were enacted during the third iteration by pilot 1.

5.1.2.1 Coding Testing

Developers tested the application in an informal way (black-box testing without documentation). There was no test report document or standard where test cases, inputs, or outputs were described. Every developer was responsible for delivering the desired functionality on time. If there was a problem that the developer could not handle, then this was taken to an internal meeting where the results from tests were discussed.


5.1.2.2 Integration Testing

The customer role was emulated with developers from other projects within the organization or market experts who were not involved during the application's development. The number of developers or the persons selected to emulate the customer depended on the manager's decisions. These emulated customers performed informal test cases that were not documented. The results of these informal test cases led to finding a defect or proposed ideas for improving the application. This report was also informal, it did not have a structure, and it was presented in an internal meeting, where decisions were made regarding the test results. Decisions depended also on managerial issues such as time to deliver and resources. Usually, if defects were found that needed to be fixed immediately, the developer responsible for this functionality needed to fix them. Otherwise, the defects should be corrected during the development of the next application version. If there were important hints for the actual application performance, then resources were assigned and a new development cycle started.

5.2 PILOT 2

5.2.1 Verification Activities

There is evidence that the following activities were enacted during the third iteration by pilot 2.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 32 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

5.2.1.1 Requirements review


The developers and the project leader met in order to review the requirements once they were specified. The following was a checklist proposed for reviewing the requirements:

- Verify that the requirement is unique: The requirement has a unique ID.
- Verify whether the requirements are testable: A procedure to test the requirement should be defined. The procedure is a strategy done using the requirements traceability matrix, where test cases are linked to a requirement.
- Verify whether the requirements are understandable: Check whether there are ambiguous requirements.
- Verify the level of detail of the requirements: The level of detail expected should not be too generic, or too detailed.
- Verify whether the requirements cover performance issues.
- Verify whether the requirements are logically organized.
- Verify whether they are not conflicting.
- Verify whether the target device is described in the requirements.

5.2.1.2 Design review

The developers and the project leader met to review their designs once they were specified. The following table shows the checklist proposed for reviewing the design:

Number	Where to look?	How to detect?	Check Box
1	All Design Diagrams	Is each name unique?	
2		Are all names used in the diagrams consistent?	
3		Are all names used in the diagrams correct?	
4		Do the design diagrams cover the system requirements?	
5	Component diagrams (Intra-Inter)	Do the components represent all interchangeable parts of the system?	
6		Are the interfaces of all components defined?	
7		Are the interfaces of all components correct?	
8		Can (and should) interfaces be simplified?	
9	Class diagrams	Are the classes consistently documented?	
10		Does each class denote a collection of similar instances?	
11	Attributes	Does each class attribute in the design class diagram have an associated data type?	
12		Are all data types primitives? If the data type is not primitive, could an association to an existing class replace it?	
13		Have initial values been specified for attributes?	
14	Associations	Is the cardinality of each association correct?	
15		Are role names given for each of the classes involved in a recursive association?	
16		Are role names given for classes that have more than one association?	
17		Are role names consistent?	

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 33 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

18		Does each association require a persistent representation? If not, could it be better modelled as an operation?	
19		Could an aggregate be better modelled with attributes?	
20		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?	
21	Sequence diagrams (intra-inter)	Is the sequence diagram consistent with the class diagram?	
22		Do the objects used in the sequence diagram belong to a class defined in the class diagram?	
23		Are all use cases mapped to a sequence diagram?	
24		Are all the variations of use cases modelled?	
25		Are the receiver objects available?	
26	Deployment diagrams	Are all the concrete components mapped to the nodes on the execution environment?	
27		Are the business model roles associated to the concrete components?	
28		Are all the relationships between the components modelled?	
29	Technology platform diagrams	Were the native platform services identified?	
30		Were the vendor-specific database services identified?	
31		Were the (server-client) services identified?	

5.2.1.3 Code review

The technical leader and the developers met in order to review the produced source code. Important issues described in the heterogeneous client document D3 were taken into account as a checklist (e.g., performance, code size).

5.2.2 Validation Activities

The following are the validation activities enacted during the third iteration of Pilot 2.


5.2.2.1 Unit Test

Developers performed unit testing in an informal way. They did not document the test cases nor the results. If a defect was found, there was no record of it, and the developer was expected to fix it immediately. The client part developers performed their unit tests using the "J2ME™ Wireless Toolkit 1.0.4". This is a client emulator that allows running midlets on top of it, and emulates the functionality of a mobile device. Specific components such as network modules were tested on cell phones, since they were critical for the rest of the application. The graphic library components were tested separately before including them in the application.

5.2.2.2 Integrate Code

Pilot 2 comprises a server part and a client part. The following were the steps taken to integrate the components of the application.

Continuous Integration of Components in the Server: The term "continuous integration" stands for nothing else than just integrating different developers' code into the system, and making sure nothing breaks. Functional requirements were tested on a separate testing suite (built in the server).

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 34 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

Integration Test for Client Components: The client part contained four modules: The communication module, the game engine module, the interface module, and the data storage module. They were tested separately and then integrated. The set of communication components were called the “transport layer”. In order to test the transport layer, a server emulator (“server stub”) had to be developed. The interface module component integration tests were performed using the “J2ME™ Wireless Toolkit 1.0.4” and were done without formal documentation, e.g., test cases or test reports.

Integrate Client and Server: Server developers performed integration tests. If a defect was found, then it was recorded in the data collection sheets defined for WP3, and communicated. The parties discussed the defect and assigned a person responsible for fixing it. Once the defect was fixed, it was declared a closed defect. These integration tests were performed when either the client or the server was not totally completed. Therefore, the principal objective was to test the communication between the client and the server.

5.2.2.3 Test Usability

A big concern of pilot developers was the use of guidelines or rules to test the usability of the user interface. As described in section 2.1, the search and evaluation of processes from related fields, was extended towards techniques for testing wireless services [77]. The activity test usability was introduced as a result of this initiative. This activity is based in Nielsen’s [89] 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, and others conform to the user needs. The activity comprises the following sessions:

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. The project manager, developers of pilot 2, and the process engineer developed the heuristics.
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.
 - 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.

5.2.2.4 Test System


The complete functionality of the system (client and server) was tested using the real environment. No emulators were used during these tests.

If a defect was found, then it was recorded in the data collection sheets defined for WP3. The parties discussed the defect and assigned a person responsible for fixing it. Once the defect was fixed, it was declared a closed defect.

6. SUMMARY AND RESULTS

This deliverable describes the procedure on how the reference process model for developing wireless Internet services was developed. The development was based on the observation of three iteration cycles and an elicitation of external knowledge (i.e., from literature) relevant to this application domain. Key principles and techniques applied were descriptive process modeling, goal-oriented elicitation, and bottom-up learning. An advantage of the presented bottom-up approach (i.e., the derivation of the reference process model from pilot processes presented in section 1.5) is the proximity of the resulting reference model to practices in real life.

The process elicitation approach used (section 3.1) has been proven effective. Descriptive process modeling helped recognize weaknesses in the process and react to them very early. Additionally, recommendations for process improvements from the developers were considered. The review of the process models was difficult because the developers were located in different geographical locations. An electronic process guide

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 35 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

(EPG) helped to review the process descriptions. Publishing the pilots' EPGs on a web site improved the review procedure.

The results from the commonality analysis (see Section 3.2) showed that it is necessary to perform it in a systematic way, in order to create a meaningful, consistent software process reference model. This assumption becomes the more relevant the larger the compared process models are. The method should be further refined. Another important finding was the usefulness of the specifically developed tool SPEARSIM for the commonality analysis. The tool does not only provide great help to the process engineer in handling complexity, but also provides assumptions based on rules. Comparing entities manually depends on the complexity and size of the process model, and on the knowledge of the process engineer. For complex process models, tool-supported comparisons may support manual analysis. Nevertheless, it is still unclear whether all available heuristics are indispensable for comparing process models.


Regarding the creation of the reference process model (see Section 3.3), it was revealed that appropriate notations for describing generic process knowledge (i.e., adaptable process models and adaptation rules) are missing. Only very specialized notations are available; appropriate graphical notations and tool support for representing them are missing. Optional product flows, for instance, are difficult to represent in existing notations.

Future research may address the following questions: How to determine effectively the impact of context factors on processes? Which degree of process complexity requires automated similarity analysis?

Summarizing the experience with the wireless Internet services domain, it can be said that the main impact factors on process design are the necessity to understand varying market demands and technology changes, as well as a set of specific non-functional requirements for wireless Internet services. Other important characteristics of the domain are: high volatility of requirements, interface usability, device independence, service scalability, seamless services, technical constraints (i.e., device memory, battery memory), difficulty to set up an appropriate test environment, and incremental planning. These characteristics were considered in the process design. Techniques, practices, and processes found to address these characteristics were integrated into the reference process model.

Testing seems to be a difficult process part for pilot partners when it comes to a wireless Internet service. Therefore, the literature search was focused on this subject. After analysing the new results of the literature search, pilot partners introduced the activity test usability for the third iteration, because they consider it the most suitable for them, and they wanted to put more emphasis on testing activities during this last iteration. The testing usability activity was performed in order to discover user interface problems or issues to improve in the developed products.

Both pilots underlined the fact that emulators available for testing wireless applications are far from being reliable. Therefore, real devices were used as early as possible in the development process.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 36 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public


APPENDIX 1. CHARACTERIZATION VECTOR DEFINITION

The characterization vector is as a tool for understanding the context of the WISE pilot projects. A characterization vector is a set of customization factors. A customization factor is a relevant characteristic that might influence the pilots' software development process. Furthermore, customization factors are the key to derive guidelines for using the software reference model. New players can identify the characteristics of their projects' context, use them as an input for the reference process model, and derive a "customized software process model" that gives the organization a more precise idea of how to face the new challenge. Although choosing the perfect set of customization factors is perhaps an impossible task due to the complexity and diversity of a project context, selecting a representative set of customization factors can be done based on the experience of the software organizations and the existing body of knowledge.

Table 8.1 presents the schema and values defined in for the wireless Internet services characterization vectors. The category schema and the definitions of the categories were taken from the "PULSE™ - Product Line Software Engineering" approach. Each customization factor has a name, a set of possible values, and reference(s).

Table 8.1: Characterization Vector definition

Categories	Customization factor	Possible values	Reference
Domain characteristics	Application type	Information system, Computation intensive system	[57]
	Wireless services domains	End user services, application domain support services, generic platform services, service management services	[58]
	Business area	Mobile online trading services, Mobile online entertainment services	WISE Project
Implementation characteristics	Project type	Client - Adaptation, Server – Adaptation, Client - New development, Server - New development	WISE Project
	Architecture	Two tier, N tier	[59]
	Application elements	Java midlets, Java servlet, JSP, Java beans, Enterprise java beans, XML, XSL, XSLT, HTML, cHTML, HDML, WML, voiceXML, WMLScript, CGI, ASP	[27], [63], [64], [24]
	Requirements specification technique	Structured text / UML cases, Intended screens	WISE Project
	Wireless networks	GSM, GPRS, Mobitex, CDPD, PDC, IMT-2000, UMTS, W-CDMA, cdmaOne, cdma2000 (MC-CDMA), Wireless Lans	[27]
	Wireless protocols	HTTP, WAP, M-Services, i-mode, Web Clipping	[27]
	Mobile devices	Desktop/Notebooks, TabletPC, PDA, HandheldPC, PDA Phones, Smart Phones, Cellular Phones.	[61]

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 37 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

Categories	Customization factor	Possible values	Reference
	Testing environment	Emulators, real devices, simulators.	[27]
	Implementation language	J2ME, J2SE, J2EE, C, WML, WMLScript	WISE Project
	Validation technique	Black box testing, white box unit testing, feature testing on target terminal	[62]
Enterprise characteristics	Organizational context	Invesnet-Italy, Motorola, Sodalìa	WISE Project
	Business objectives	Capturing knowledge, producing high quality products	[57]
	Role	Service provider, content provider, technology provider, service developer	[58]
	Experience in software development	5<X<10 years, > 10 years	WISE Project

CATEGORIES

The customization factors are grouped into the following categories [57]:

- Domain Characteristics: factors that relate to the domain itself and are independent of implementation aspects.
- Implementation Characteristics: factors that influence implementation in the domain.
- Enterprise Characteristics: factors relating to the enterprise in general.

DOMAIN CHARACTERISTICS

The following is the description of the customization factors that belong to the domain characteristics category:

Application type: This customization factor identifies the main aspect that dominates the complexity of the application. The defined values are:


- Information management systems (data/object view)
- Computation-intensive systems (transformational/functional view) (e.g., simulators).
- Control-intensive systems (behavioral view)

Reference: This customization factor and its values were taken from the [57] document.

Wireless Services Domains: The customization factor identifies five categories of domains for the services at different levels. The domains are described in the following table:

Table 8.2. Wireless Services Domains

Domain	Description
End-user services	Services that are provided directly to the end users (i.e., customers)
Application domain support services	Services that provide generic services for a specific application domain on which end user applications rely, but not usually provided for the end user as such (e.g., a game engine)
Generic platform services	Services that are needed for the end-user applications and application support services, but they are not directly related to any application domain (e.g., instant messaging API or a GPS location)

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 38 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

	service).
Technology platform services	Services related to specific implementation technology (software or hardware) choices either in mobile terminal or on server side (e.g., HTTP, Web browser, mobile phone, Java).
Service management	Services that are needed to make the service available and link it to business processes. These are not directly related to the purpose of end-user applications (e.g., service downloading).

Reference: This customization factor and its values were taken from [58].

Business Area: These are the real world entities that are used to describe the domain.

Reference: This customization factor was taken from [57]. The values were defined by the WISE project.

IMPLEMENTATION CHARACTERISTICS

The following is the description of the customization factors that belong to the implementation characteristics category:

Project Type: This refers to the nature of the implementation activities to be performed. The defined values are:

- Client – Adaptation: Client software is modified due to new requirements. These new requirements could be generated due to a defect or new functionality.
- Server – Adaptation: Server software is modified due to new requirements. These new requirements could be generated due to a defect or new functionality.
- Client – New: Client software is created from scratch.
- Server – New: Server software is created from scratch.

Reference: This customization factor and its values were defined by the WISE project.

Architecture: This refers to the logical structure of the application. The defined values are:

- Two-tier: An application running on the client (i.e., a web browser or a java/J2ME client) connects the web server for a request. The web server sends a response to the client.
- N-tier: The client (i.e., a browser or java/J2ME client) connects to an application server. The application server contains the business logic, and the actual application. The database server contains the relevant data. Middleware software handles the communication, distribution of processes, and data translation, between the client and the application server and between the application server and the data server.


Reference: This customization factor and its values were taken from [59].

Application Elements: An application environment consists of different components. The focus is set on the Java elements and the different markup languages for the mobile devices. The defined values are:

- Java midlets: "Midlets" is the name for Java technology applications that run on wireless and mobile devices. Midlets are written to the Mobile Information Device Profile (MIDP) in the Java 2 Platform, Micro Edition (J2ME).

Reference: This concept was taken from [63].

- Java servlets: Java servlets are used to provide dynamic Web content based on an HTTP request. In general, they are server-side Java programs. Therefore, they offer all the advantages of the Java language.
- JavaServer Pages (JSPs): JSPs are used to generate the formatted output for a Web page response. JSPs do not include any business logic. Therefore, they are useful to separate the development of the Web site from the Web page design.
- JavaBeans: JavaBeans are simple component classes in the Java language and are mainly used as reusable objects.
- Enterprise Java Beans: The main focus and advantages of Enterprise Java Beans (EJBs) are to separate the business logic of an application from the middleware supporting it.
- XML: XML stands for eXtensible Markup Language. It is a subset of the SGML standard and is also used to define markup languages like WML or SyncML. XML is an easy to use language for

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 39 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

describing any content. Developers can define their own markup language tags or elements. They can ensure that their XML dialect best fits their application needs. In general, XML documents include only data. They do not include any formatting information used to present the content; this information is stored in a separate Extensible StyleSheet Language (XSL) file.

- The Extensible Stylesheet Language (XSL): XSL is used for the presentation of an XML document. XSL produces various output documents, for example, HTML,WML, cHTML and VoiceXML.
- Extensible Stylesheet Language Transformation (XSLT): XSLT is the common language to transform one XML document into another output format by applying the appropriate StyleSheet for it.
- The Hypertext Markup Language (HTML): HTML is the common language for creating and publishing documents for the World Wide Web. It is based on SGML and used to display hypertext information on traditional Web browsers.
- Compact HTML (cHTML): cHTML was developed and announced in 1999 by the Japanese telecom provider NTT DoCoMo. It was specified to fit the new requirements for small wireless devices like, for example, mobile phones. cHTML is actually the markup language used for the i-mode standard.
- Handheld Device Markup Language (HDML): HDML is a specialized version of HTML. Designed to enable wireless pagers, cell phones, mobile phones and other handheld devices to obtain information from Web pages.
- The Wireless Markup Language (WML): WML is defined by the WAP Forum. Like cHTML, it was developed to meet the requirements of small appliances like mobile phones or PDAs; WML is based on XML. The content of a WML file can be viewed on any WAP browser. Unfortunately, as in HTML, the visual presentation of a WML document depends on which browser it is displayed on.
- WMLScript: WMLScript is a scripting language used to program the mobile device. WMLScript is an extended subset of the JavaScript™ scripting language.
Reference: This concept was taken from [64].
- Extensible HTML (XHTML): XHTML is a combination of HTML 4.0 and XML 1.0 into a single format for the Web. XHTML is expected to become the standard format for Web pages. XHTML also makes it possible for Web pages to be developed with different sets of data, depending on the type of browser used to access the Web.
- The Voice Extensible Markup Language (VoiceXML): VoiceXML is based on XML. It is used for creating distributed voice applications based on existing or new Web applications. Those applications can be accessed by the user by telephone. It is also possible to refer to VoiceXML as a “non-visual” markup language because the output is not presented to the user visually.
- CGI: The common gateway interface was one of the first methods for creating web applications. It defines a communication standard between a CGI application and the web server. As it is a communication protocol rather than a language, CGI applications can be written with a variety of languages including C, C++, Java, Perl or Visual Basic. Perl, in particular, is a popular choice as it is simple, high-level and has many freely available extension libraries.
Reference: This concept was taken from [24].
- ASP: *Active Server Pages (ASP)s* has been developed by Microsoft and can use JScript, Perl or VBScript as a scripting language. On the Windows Platforms ASP, is based on COM, Microsoft's Component Object Model, so scripts can call COM components in a similar way to JSP calling Java Beans. The COM architecture is acknowledged as being rather complicated and slow. ASP also supports easy construction of database applications via Microsoft's Active Data Objects and SQL.
Reference: This concept was taken from [24].


Reference: This customization factor and most of its values are taken from [27]. The following values were taken from other sources (Java midlets, WMLScript, CGI, ASP).

Requirements Specification Techniques: The defined values are:


- Structured text / UML cases: A template is used for specifying the requirements. The requirements are specified with UML use cases and text.

Reference: This customization factor and its values were defined by the WISE project.

Wireless Networks: Wireless networks are used to transmit data between mobile devices or personal computers using wireless adapters without the use of a physical cable or wire.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 40 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- GSM: The standard was proposed in 1982 and completed in 1990, while the first networks were deployed in 1991. The main reason behind the introduction of GSM in Europe was to provide a common standard for European Cellular Communications, which allowed subscribers to roam throughout Europe and access cellular networks in each country with the same equipment. Today, GSM is the most important and widespread mobile standard worldwide. Many variants of GSM have been created for different frequency ranges. GSM technology in the form of DCS1900 is available in North America, and often referred to as Personal Communications Services (PCS) systems. In a GSM network, the subscriber is considered an entity separate from the device. This means that the subscriber identity can be transferred from one physical phone to another, without reprogramming the device. This is accomplished by means of a Subscriber Identity Module (SIM), which is a small smartcard to insert into the mobile phone.
- GSM: General Packet Radio Service (GPRS) is the packet-based extension of the GSM network. PRS is a fundamental step in the migration from GSM to 3G networks. GPRS can support data traffic over packet-based connections with a higher bit rate than GSM (up to 172.4 kbps). GPRS introduces three additional coding schemes (CS) for the data that is transmitted across the radio interface, with respect to the single coding scheme existing in GSM. These coding schemes provide different degrees of error correction and, consequently, different bandwidths.
- Mobitex: The Mobitex technology was originally developed in Sweden in 1984. This network now operates in 23 countries. Mobitex is the network used in the U.S. by Palm.Net, as well as by many other wireless service providers. The data rate of a Mobitex wireless channel is 8 kbps. The network latency is relatively high and varies significantly. Both mobile devices and fixed terminals are treated equally in terms of addressing. Any entity can communicate with every end system in the Mobitex network. It is even possible to address end systems at other network providers if they are interconnected.
- CDPD: Cellular Digital Packet Data (CDPD) is a wireless, packet-switched network technology. It was built on top of the AMPS infrastructure by adding the required capabilities for packet management and routing. Roughly speaking, CDPD is the packet-based extension of AMPS, which is circuit-switched, exactly as GPRS is the packet-based extension of GSM. CDPD inherently uses the Internet Protocol (IP) as the protocol for sending and receiving data. IP includes protocols that take care of such essential functions as authentication and encryption, and provides a maximum raw data throughput of 19.2 kbps.
- Personal Digital Cellular (PDC) is a Japanese cell phone standard. The data transmission rate is 9.6 kbps. Almost all the cell phones used in Japan are based on PDC.
- IMT-2000: International Mobile Telecommunications-2000 (IMT-2000) is a 3G wireless system. IMT-2000 offers support for a wide range of mobile devices, includes links to terrestrial and/or satellite-based networks, and the terminals may be designed for mobile or fixed use.
- UMTS: The Universal Mobile Telecommunications System (UMTS) is the European implementation of the 3G wireless phone system. UMTS, which is part of IMT-2000, offers global roaming and personalized features. UMTS was designed as an evolutionary system for GSM network operators, and offers impressive data rates of up to 2 Mbps. UMTS uses the W-CDMA technology. GPRS and EDGE are interim steps that will speed up wireless data for GSM.
- The Direct Spread - Code Division Multiple Access (DS-CDMA) specification is supported by Ericsson (Sweden) and Nokia (Finland). The data translation rate is 64 kbps for upstream and 384 kbps for downstream.
- CdmaOne: Code Division Multiple Access (CDMA) is a specification of wireless communication. Voices from multiple users are transformed by multiplying different codes and transferred all together as one frequency. The receiver can detect only the sender's voice and decode it. The cdmaOne is one standard of 3G cell phones that uses the CDMA protocol. The data transmission rate is 14.4 kbps.
- cdma2000 (MC-CDMA): The Multi Carrier - Code Division Multiple Access (MS-CDMA) specification is supported by Qualcomm (US) and Lucent Technologies (US) and will be the North American standard. The maximum data translation rate will be 14.4 kbps while fast moving, 384 kbps while slow moving, and 2 Mbps while at a standstill.

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 41 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- Wireless LANs: A wireless LAN is a local area network that transmits over the air, typically using an unlicensed frequency such as the 2.4 GHz band. Wireless access points (base stations) are connected to an Ethernet hub or server and transmit a radio frequency over an area of several hundred to a thousand feet. This frequency can penetrate walls and other non-metal barriers. Roaming users can be handed off from one access point to another like a cellular phone system.

Reference: This customization factor and its values are taken from [27].


Wireless Protocols: Wireless protocols are used to connect mobile devices to the Internet. Many of the wireless protocols have defined architectures that optimize the use of the radio resource and also minimize the capabilities required for the device.

- HTTP Protocol: This approach, which is very suitable for wired connections, presents many limitations when adopted in wireless networks. In fact, when using TCP/IP based protocols (such as HTTP) in wireless mobile networks, you have to deal with the fact that radio networks usually have much less bandwidth and a higher latency than local or wide area networks. Moreover, wireless connections are less stable in nature than wired connections and also unpredictable in terms of availability. TCP/IP-based protocols work well over wireless connections. However, performance is slow.
- The Wireless Application Protocol (WAP) is a standard for providing cellular phones, pagers, and other handheld devices with secure access to e-mail and text-based Web pages. WAP features the Wireless Markup Language (WML), which was derived from Phone.com's HDML and is a streamlined version of HTML for small-screen displays. It also uses WMLScript, a compact JavaScript-like language that runs on limited memory. WAP also supports handheld input methods such as a keypad and voice recognition. Independent of the air interface, WAP runs in all the major wireless networks in place now and in the future.
- M-Services: Mobile Services (M-services) is a new initiative from the WAP Forum. After a couple of years of developing the Wireless Application Protocol, the Forum has started to work out a new protocol to replace WAP. This time not only the development reached a new version of WAP technology, but the upcoming communication technologies require new services and support on the application level. M-services bring new functions and features to the mobile devices on the application level, for example: enhanced application support for the devices; advanced GUI and peripheral support; modular software.
- I-mode: i-mode is a wireless service developed by NTT DoCoMo in Japan. It is designed to provide mobile phone voice service, Internet and e-mail access. The i-mode protocol uses compact HTML (cHTML) as its markup language for the reasons that WAP use WML.
- WebClipping: Web Clipping is a proprietary technology developed by Palm. The main elements that constitute the Web Clipping architecture are the Palm device, a Web Clipping proxy server, and the content server. In order to support Web Clipping, a special piece of software called a Clipper must be present on the Palm device.

Reference: This customization factor and its values are taken from [27].

Mobile Devices: Devices that have as their main characteristic the possibility to connect them to the Internet. Defined values are:

- Desktops / Notebooks: These are normal PCs with an integrated card for wireless networks. Processing speed is high.
- TabletPCs: TabletPCs often have a pointing device as a touch screen. TabletPCs without such pointing device are most commonly called Subnotebooks. Processor speed today ranges from a 200Mhz ARM processor to full last generation Pentium III/IV, and memory sizes vary accordingly. Operating systems can be those of desktops or those of PDAs, depending on device.
- PDAs: entirely based on a touch screen the most common components of this family are Palm PC and PocketPC. Palm PCs are developed by the Palm and Visor companies and are based on the Motorola Dragonball processor (up until now, at least) at a typical clock rate of no more than 20MHz. Memory size can be 16, 32, 64 MB. Palm OS is the referring Operating System (many versions available). Memory size ranges from 1MB to 16MB. PocketPCs are developed by many companies, are based on ARM (mostly Intel StrongARM) processors with a clock frequency that ranges from 150MHz to 206 MHz and above. They are all based on Microsoft PocketPC operating system

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 42 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

(basically a downgrade of the common desktop Microsoft OS Windows NT) and the vast majority now has a color display.

- Handheld PCs: basically PDAs with also a small keyboard.
- PDA phones: an extension to PDA where connectivity to the phone network is deeply integrated in the device and not added.
- Smart phones: same as PDA phones but coming from a cellular world rather than from the computer world.
- Cellular phones (J2ME enabled): the electronic best sellers of those last few years have been enhanced to support the download of applications and services. Today it is common to find J2ME enabled phones, but in the near future, other programming platforms may be available (i.e., Qualcomm is also porting BREW on non-CDMA platforms).

Reference: This customization factor and its values are taken from [61].

Testing Environment:

- Real devices: The real devices, i.e., the mobile clients, are the same ones as those that will work in the runtime environment after implementation. They are the best for development and testing. The problem with real devices is that they are expensive and usually require additional infrastructure, which makes them more expensive. Furthermore, real devices can be slower than emulators, because of the real environment.
- Emulators: Software equivalent to the original device. The wireless emulators run on a desktop client. The emulators are give the same user experience on the screen as real devices. Additionally, they are emulating the network connection to the Internet via the operating system's network connection.
- Simulators: Special devices where the simulating device copies the behavior of the original device. Simulators are expensive devices; they are best for hardware development.

Reference: This customization factor and its values are taken from [27].

Implementation Language: This refers to the programming languages used for implementing the application. Defined values are:

- J2ME: The Java™ 2 Platform, Micro Edition (J2ME™) is the Java platform for consumer and embedded devices such as mobile phones, PDAs, TV set-top boxes, in-vehicle telematics systems, and a broad range of embedded devices.
- J2SE: J2SE™ provides the essential compiler, tools, runtimes, and APIs for writing, deploying, and running applets and applications in the Java programming language.
- J2EE: J2EE™ technology and its component-based model simplify enterprise development and deployment. The J2EE platform manages the infrastructure and supports the Web services to enable development of secure, robust and interoperable business applications. The J2EE platform is the foundation technology of the Sun ONE platform and Sun's Web services strategy.
- C: General purpose programming language.

Reference: This customization factor was defined by the WISE project. The value definitions are taken from the website www.sun.com.


Validation Technique: This refers to the methodologies and techniques used to test the application. Defined values are:

- Black box testing: relies on the specification of the system or component that is being tested to derive test cases. The system is a black box whose behavior can only be determined by studying its inputs and outputs.
- White box testing: The testers analyze the code and use the knowledge about the structure of a component to derive the data.
- Feature testing: The testers analyze the code that implements specific features of the system.

Reference: This customization factor and its values are taken from [62].

ENTERPRISE CHARACTERISTICS

The following is the description of the customization factors that belong to the enterprise characteristics category:

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 43 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

Organizational Context: This refers to general information about the development organizations. The actual values are:

- Motorola: Motorola is involved in Italy in the mobile, data communications and semiconductors areas. 300 employees work in three major offices located in Rome, Turin and Milan. Motorola has a well-established department of Software Process Management, which bases its activity on the CMM.
- Investnet: Investnet offers online trading services to high-end customers, such as banks and brokers, that require quasi 100% availability, 100% reliability of data, and immediate response time. The company is a leader in this very selective market in Europe thanks to the technological platform used, RealTick, and to its focus on innovation.

Reference: This customization factor and its values were defined by the WISE project.

Business Objectives: This factor describes the business goals of the software development organizations. They are:

- Capturing knowledge: By capturing the domain technical knowledge and methodological development knowledge, the software development organizations will have the opportunity to join the market and/or remain competitive.
- Producing high quality products: The execution of the project aims at learning and applying practices that will guarantee high quality products.

Reference: This customization factor and its values were defined by the WISE project.


Roles: This factor describes the way software organizations are involved in the wireless Internet Services business. The defined values are:

- Wireless Access Provider: The wireless access provider is responsible for providing access to one or more public wide-area wireless network services, which in turn provide the pipes through which applications and information flow to wireless devices.
- Service Provider: The service provider is responsible for delivering a managed service to business customers, to fulfill part or all of a customer's wireless enterprise solution.
- Application Provider: The application provider typically sells packaged software products that automate business processes to large or small business customers.
- Technology Provider: The technology provider is responsible for manufacturing, branding, and supporting mobile devices (laptops, PDAs or other data-capable handsets).
- Mobile Middleware Provider: The mobile middleware provider is responsible for developing and supporting software infrastructure products, which do one or more of the following:
 - o make it easier to develop mobile extensions to existing applications, or new mobile applications.
 - o improve the runtime quality of service experienced by application users.
 - o provide a wireless bridge, which knits together the network protocols used by wireless access network services and the network protocols used by corporate networks.
 - o allow IT or network staff to remotely monitor and manage the new technology elements that one or more deployed mobile applications introduce.
- Solution Builder: The solution builder is responsible for assembling products and services provided by other roles in the value web, and carrying out custom integration work in order to build a complete solution for the customer.
- Content Providers: The content providers sell their content to portals, service-providers and customers.
- Network Operators: The network operators sell network capacity to consumers via service providers. At the moment the main source of revenue for network operators is the voice, but as mobile data services increase popularity, operators will have to cooperate with portals and content providers to succeed in the wireless business.

Reference: This customization factor and its values were defined by [66].


Experience in Software Development: This factor describes the experience of the developer team. The values are given as the average number of years that the team in charge of the project has been working developing software products.

Reference: This customization factor and its values were defined by the WISE project.


	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 44 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

REFERENCES


- [1] Cockburn, A.: Agile Software Development: Addison-Wesley Pub Co; ISBN: 0201699699; 1st edition (2001).
- [2] Friday, A., Davies, N., Blair, G.S., Cheverest, K.W.J.: Developing Adaptive Applications: The MOST Experience. Integrated Computer Aided Engineering: ICAE, vol 6, No 2, pp. 143-158 (1999).
- [3] Nilsson, A., Anselmsson, M., Olsson, K., Johansson, Erik.: Impacts of Measurement on an SPI Program. Q-Labs (http://www.q-labs.com/files/Papers/SPI99_Imp_of_Meas_on_SPI.pdf).
- [4] Roe, C., Gonik, S.: Server-Side Design Principles for Scalable Internet Systems. IEEE Software, vol.19, No. 2, pp. 34-41 (2002).
- [5] Wong, C., Chu, H., Katagiri M: A Single Authoring Technique for Building Device Independent Presentations. In: W3C Device Independent Authoring Techniques Workshop, (2002).
- [6] Adamopoulos, D.X., Pavlou, G., Papandreou, C.A.: An Integrated and Systematic Approach for the Development of Telematic Services in Heterogeneous Distributed Platforms. Computer Communications, vol. 24, pp. 294-315 (2001).
- [7] Raffo, D., Kaltio, T., Partridge, D., Phalp, K., Ramil, J.F.: Empirical Studies Applied to Software Process Models. In: International Journal on Empirical Software Engineering, vol. 4, No. 4 (1999).
- [8] Schwabe, D., Mattos, G.R., Rossi, G.: Cohesive Design of Personalized Web Applications. IEEE Internet Computing vol. 6, No 2, pp 34-43 (2002).
- [9] Chi, E.: Improving Web Usability Through Visualization. IEEE Internet Computing, vol. 6, No 2, pp. 64-71 (2002).
- [10] Karlsson, E., Taxen, L.: Incremental Development for AXE 10. ACM SIGSOFT Software Engineering Notes, vol. 22, No. 6 (1997).
- [11] Karlsson, E.: A Construction Planning Process. Q-Labs, LD/QLS 96:0381, Lund Sweden (1999).
- [12] Brooks, F.P. Jr.: The Mythical Man-Month. Essays on Software Engineering, Anniversary edition. Addison Wesley. Reading MA (1995).
- [13] Giannetti, F.: Device Independency Web Application Framework. In: W3C Device Independent Authoring Techniques Workshop, 25-26 (2002).
- [14] McGarry, F., Pajerski, R., Page, G., Waligora, S., Basili, V.R., Zelkowitz, M.V.: An Overview of the Software Engineering Laboratory. Software Engineering Laboratory Series Report, SEL-94-005, Greenbelt MD USA (1994).
- [15] Mori, G., Paterno, F., Santono, C.: An XML Based Approach for Designing Nomadic Applications. In: W3C Device Independent Authoring Techniques Workshop, (2002).
- [16] Omidyar, G.: Internet Services over Mobile and Wireless Networks Architectures and Protocols. In: Conference Proceedings of the First European Conference on Universal Multiservice Networks, pp, 1-4, France (2000).
- [17] Cooper, H.M.: Scientific Guidelines for Conducting Integrative Research Views. Review of Educational Research, vol. 52, No. 2, pp.291-302.
- [18] Rombach, H.D., Verlage, M: Directions in Software Process Research. Advances in Computers, vol. 41, pp. 1-63 (1995).
- [19] Zettel, J., Maurer, M., Münch, J., Wong, L.: LIPE: A Lightweight Process for E-Business Startup Companies based on Extreme Programming. Proceedings of the Third International Conference on Product-Focused Software Processes Improvement (PROFES), pp. 255-270, (2001).
- [20] Basu, K., Hee, Lee.: Challenge of Universal Mobility and Wireless Internet. Conference Records of the 2000 IEEE Wireless Communications and Networking Conference, pp. 1563-1568 (2000).
- [21] Beck, K.: Extreme Programming Explained: Embrace Change. Addison Wesley, (2000).
- [22] Constantine, L.L., Lockwood, A.D.L.: Usage-Centered Engineering for Web Applications. IEEE Software, vol. 19, No. 2, pp.42-50 (2002).
- [23] 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).
- [24] Buttler, M.H.: Current Technologies for Device Independence. HP Laboratories, HP-2001-83, Bristol, (2001).

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 45 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- [25] Hammar, C.M.: Designing User-Centered Web Applications in Web Time. IEEE Software, vol. 18, No. 1, pp. 62-69 (2001).
- [26] 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).
- [27] 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).
- [28] Becker-Kornstaedt, U., Hamann, D., Kempkens, R., Rösch, P., Verlage, M., Webby, R., Zettel, J.: Support for the Process Engineer: The Spearmint Approach to Software Process Definition and Process Guidance. Proceedings of the Eleventh Conference on Advanced Information Systems Engineering (CAISE '99), pp. 119-133. Lecture Notes in Computer Science, Springer-Verlag. Berlin Heidelberg New York (1999).
- [29] Becker-Kornstaedt, U., Boggio, D., Muench, J., Ocampo, A., Palladino, Gino.: Empirically Driven Design of Software Development Processes for Wireless Internet Services. Proceedings of the Fourth International Conference on Product-Focused Software Processes Improvement (PROFES), (2002).
- [30] Nerurkar, U.: Web User Interface Design: Forgotten Lessons. IEEE Software, vol. 18, No. 6, pp. 69-71 (2001).
- [31] Yau, V.: Project Management Strategies and Practices for Wireless CDMA Software Development. Proceedings of the IEEE International Conference on Industrial Technology, (1996).
- [32] <http://www.w3.org/TR/2001/WD-di-princ-20010918/>
- [33] 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).
- [34] Odlyzko, A.: Paris Metro Pricing: The Minimalist Differentiated Services Solution. AT&T Laboratories Research, April 1999.
- [35] Maurer, F., Martel, S.: Rapid Development for Web-Based Applications. IEEE Internet Computing, vol 6, No 1, pp. 86-90 (2002).
- [36] Boehm, B.W.: A Spiral Model for Software Development and Enhancement, IEEE Computer, vol 21, No 5, pp. 61-72 (1988).
- [37] Boehm, B.W.: Get Ready for Agile Methods, with Care, IEEE Computer, vol 35, No 1, pp. 64-69 (2002).
- [38] McDermid, J.A., Rook, P.: Software Development Process Models, Software Engineer's Reference Book, Ed., Boca Raton, FL: CRC Press, pp. 15.26 – 15.28. (1994).
- [39] Wong, C., Chu, H., Katagiri, M.: GUI Migration Across Heterogeneous Java Profiles. Proceedings of ACM SIGCHI-NZ'02. (2002).
- [40] Rosenfeld, L., Morville, P.: Information Architecture for the World Wide Web, O'Reilly, New York, (1998)
- [41] Highsmith, J., Cockburn, A.: Agile Software Development: The Business of Innovation, Computer, vol 34, No 9, pp. 120-122. (2001).
- [42] Cockburn, A., Highsmith, J.: Agile Software Development: The People Factor, Computer, vol 34, No 11, pp. 131-133. (2001).
- [43] Read, K., Maurer, F.: Developing Mobile Wireless Applications. IEEE Internet Computing, vol 7, No 1, pp. 81-86. (2003).
- [44] <http://wireless.java.sun.com/midp/tips/appsize/>
- [45] <http://java.sun.com/j2me/docs/alt-html/midp-style-guide7/preface.html>
- [46] Zheng, P, Ni, M.N: EMPOWER: A Network Emulator for Wireless and Wireline Networks. In Proceedings of IEEE INFOCOM (2003).
- [47] 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).
- [48] Carlshamare, P.: Release Planning in Market-Driven Software Product Development: Provoking and Understanding. Requirements Engineering, No. 7, pp. 139-151, (2002).
- [49] Ritel, H., Webber, M.: Planning problems are wicked problems. In: Cross N (ed). Developments in Design Methodology. Wiley, Chichester, pp 135-144, (1984).
- [50] 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).
- [51] Buchanan, G., Farrant, S., Jones, M., Thimbleby, H., Marsden, G., Pazzani, M.J.: Improving Mobile Internet Usability. In proceedings World Wide Web 10, pp. 673-680, (2001).

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 46 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- [52] Nielsen, J.: Graceful Degradation of Scalable Internet Services, WAP: Wrong Approach to Portability, Alertbox 31/10/1999 at <http://www.useit.com/alertbox/991031.html>
- [53] Nielsen, J.: WAP Backslash, Alertbox 09/07/2000 at <http://www.useit.com/alertbox/20000709.html>
- [54] Brinkkemper, S., Harmsen, F., Oei, H.: Configuration of Situational **Process Models**: An Information Systems Engineering Perspective. In W. Schaefer (ed.) Proceedings of the European Workshop on Software Process Technology (EWSPT'95), LNCS 913. Berlin: Springer-Verlag, 193-196. (1995)
- [55] Highsmith, J.: What is Agile Software Development? The Journal of Defense Software Engineering, October, (2002).
- [56] DSDM Consortium: Dynamic Systems Development Method. Version 3, <http://www.dsdm.org>.
- [57] Bayer J., Flege O., Knauber P., Laqua, R., Muthig, D., Schmid, K., Widen, T.: PULSE™- Product Line Software Engineering. IESE-Report No.020.99/E (1999).
- [58] Kalaoja, J., Niemelä, E., Tikkala, A, Kallio, P.: WISE Knowledge Base and Reference Architecture. Deliverable D4B. Wireless Internet Service Engineering. IST-2000-30028.
- [59] Offutt, J.: Qualitative Attributes of Web Software Applications. IEEE Software, vol 19, no.2, pp. 25-32. (2002).
- [60] Conallen, J.: Building Web Applications with UML. Addison-Wesley 2000.
- [61] Forchino, F., Negro, M.: Management of Heterogeneous Clients. Deliverable D3. Wireless Internet Service Engineering. IST-2000-30028.
- [62] Sommerville, I.: Software Engineering. Addison-Wesley 2001.
- [63] Hardy, M.: The Magic of Midlets. A Pocket Guide to the Java 2 Platform, Micro Edition (J2ME Technology). Java One Conference 2001. <http://java.sun.com/features/2001/06/golden.j2me.html>.
- [64] WAP Forum, Ltd: WMLScript Specification. <http://www1.wapforum.org/tech/documents/WAP-193-WMLScript-20001025-a.pdf>. (2001)
- [65] Webby, R., Becker, U.: Towards a Logical Schema Integrating Software Process Modeling and Software Measurement. In: Harrison, R. (ed.): Proceedings of the Nineteenth International Conference on Software Engineering Workshop. Process Modeling and Empirical Studies of Software Evaluation, pp. 84-88 Boston USA (1997)
- [66] Kallio, P: Business Models In Wireless Internet Service Engineering. Deliverable D1. Wireless Internet Service Engineering. IST-2000-30028.
- [67] Verlage, M.: An Approach for Modeling Large Software Development Processes by Integrating Role Specific Views Modeled Independently. PhD thesis, Fachbereich Informatik, University of Kaiserslautern, Germany, (1997). (In German).
- [68] Maiden, N., Ncube, C.: Acquiring COTS Software Selection Requirements. IEEE Software: 46-56, March/April (1998).
- [69] 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).
- [70] Ochs, M.A., Pfahl, D., Chrobok-Diening, G.: A Method for Efficient Measurement-based COTS Assessment and Selection - Method Description and Evaluation Results. Proc. of IEEE 7th International Software Metrics Symposium, pp. 285-296. (2001).
- [71] Morisio, M., Tsoukiàs, A.: lusWare: A methodology for the evaluation and selection of software products, *IEE Proceedings-Software*, 144 (3): 162-174. (1997).
- [72] Feblowitz, D.M., Greenspan, S. J.: Scenario-Based Analysis of COTS Acquisition Impacts. *Requirements Engineering*, 3 (3/4): 182-201 (1998).
- [73] Lawlis, P.K., Thomas, D.M., Courtheyn, T.: A Formal Process for Evaluating COTS Software Products, *IEEE Computer*, 34 (5): 58-63, (2001).
- [74] 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).
- [75] Humphrey, W.S.: Introduction to the Team Software Process. SEI Series In Software Engineering. Addison Wesley. (1999).
- [76] Briand, L.C., Differding, C., Rombach, H.D.: Practical Guidelines for Measurement-Based Process Improvement. Software Process. Improvement and Practice, vol. 2, No. 4, pp. 253-280 (1996)

	Service Engineering Process (Empirical approach for creating the reference process model) Deliverable ID: D2 (Part A)	Page : 47 of 47
		Version: 03.05 Date: 17 Sep 04
		Status : Final Confid : Public

- [77] Ganesan, S., Ocampo, A., Posega, M: Techniques for Validating Wireless Internet Services. IESE Report (147.03/E). (2003).
- [78] Becker, A.S.: A Usability Perspective on Wireless Internet Technology, Computer Science & Software Engineering, Florida Institute of Technology, (2001).
- [79] Thimbleby H., Cairns, P., Jones M.: Usability Analysis with Markov Models, ACM Transactions on Computer-Human Interaction, Vol. 8, (2001).
- [80] Offutt, A.J., Liu, S., Abdurazik A.: Generating Test Data From State-based Specifications, The Journal of Software Testing, Verification, and Reliability, Wiley, vol. 13, No.1, pp, 25-53, (2003).
- [81] 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).
- [82] Rizzo, L.: Dummynet: a simple approach to the evaluation of network protocols, proceedings of ACM Computer Communication Review, vol. 27, (1997).
- [83] Fall, K.: Network Emulation in the VINT/NS Simulator, In Proceedings of 4th IEEE Symposium on Computers and Communications, (1999).
- [84] Noble, B.D., Satyanarayanan, M., Gao, 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.
- [85] 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).
- [86] Puuskari, M.: General Packet Radio Services for GSM, University of Helsinki Department of Computer Science Spring (2000).
- [87] Jain, A, Tyago, S.: Simulation Testing in Mobile Networks – Protocols and Applications Perspective, Telecom & Networking Division, HCL Technologies Ltd.
- [88] Sinha, A., Chandrakasan, A.: JouleTrack - A Web Based Tool for Software Energy Profiling, Proceedings of the 38th Design Automation Conference, (2001).
- [89] Nielsen, J., Mack, R.L.: Usability Inspection Methods / John Wiley & Sons, Inc; (1994).