UNIVERSITÀ DEGLI STUDI DI ROMA "LA SAPIENZA"

FACOLTÀ DI INGEGNERIA

Tesi di Laurea Specialistica in

INGEGNERIA INFORMATICA

Metodologie User-Centered. L'Approccio ed il Caso di Studio del Progetto WORKPAD

Relatore

Ing. Massimo Mecella

Candidato

Andrea Marrella

Anno Accademico 2008/2009

UNIVERSITÀ DEGLI STUDI DI ROMA "LA SAPIENZA"

FACOLTÀ DI INGEGNERIA

Tesi di Laurea Specialistica in

INGEGNERIA INFORMATICA

Metodologie User-Centered. L'Approccio ed il Caso di Studio del Progetto WORKPAD

[User-Centered Design Methodologies. The Approach and the Case of the WORKPAD Project]

Relatore

Ing. Massimo Mecella

Candidato

Andrea Marrella

Anno Accademico 2008/2009

AUTHOR'S ADDRESS:

Andrea Marrella
Via Edoardo Garbin 47, I-00139 Rome, Italy
E-MAIL: a.marrella@gmail.com

Questa tesi è principalmente dedicata alla mia famiglia,
che mi ha dato la grande opportunità di affrontare questo
cammino mostrandomi sempre un sostegno impagabile,
e a Chiara, che non ha mai mancato di incoraggiarmi
e mi è stata vicina soprattutto nei momenti più difficili.
Un ringraziamento speciale va a Massimo, grazie al quale
ho acquisito un'esperienza lavorativa
ed umana più unica che rara.

Grazie di cuore.

Extended Abstract

Lo sviluppo tecnologico che negli ultimi anni ha interessato dispositivi mobili quali Smartphone e PDA (Personal Digital Assistant) ha contribuito alla diffusione di tali dispositivi in diversi settori applicativi. Attualmente i dispositivi mobili rappresentano una nuova piattaforma per lo sviluppo di applicazioni innovative in grado di sfruttare appieno le caratteristiche di dinamicità che l'ambiente mobile offre. La sempre maggior disponibilità di dispositivi in grado di accedere alle moderne reti wireless (Wi-Fi ed UMTS), associata ad un notevole incremento delle capacità computazionali, apre nuovi scenari applicativi dominati dalla mobilità. Le caratteristiche di connettività sempre più sofisticate consentono a dispositivi dotati di adattatori wireless di costruire reti ad-hoc (MANET) che assicurano la possibilità di comunicare anche senza il supporto di tradizionali infrastrutture di rete. L'elevata dinamicità che caratterizza le reti ad-hoc ne favorisce l'utilizzo in tutti quei settori che richiedono l'esecuzione di attività di tipo cooperativo.

In particolare, negli ultimi anni è emersa la necessità di sistemi software che forniscano supporto e facilitino la collaborazione tra diverse entità che cooperano nell'esecuzione di attività complesse. A tal proposito, si consideri l'insieme di attività che devono essere eseguite nella gestione di complessi scenari d'emergenza. In seguito ad eventi catastrofici e calamità naturali è necessario pianificare, eseguire e coordinare una serie di attività allo scopo di fornire assistenza alle vittime e stabilizzare la situazione, riducendo la probabilità di danni secondari e velocizzando le attività di recupero. In tale ambito di gestione di scenari di emergenza opera il progetto WORKPAD[1], finanziato dalla Comunità Europea.

Il progetto WORKPAD mira alla costruzione e allo sviluppo di una innovativa struttura software di comunicazione orientata ai servizi, come supporto collaborativo al lavoro degli operatori civili negli scenari di emergenza. In tali scenari, diverse squadre, appartenenti a svariate organizzazioni, hanno necessità di collaborare tra loro per raggiungere il medesimo risultato. Ogni team ha il compito di eseguire un insieme di interventi, volti a fronteggiare l'emergenza. Differenti team (di differenti organizzazioni) collaborano al fine di raggiungere un macro-obiettivo, ognuno svolgendo le attività di propria competenza. Ciascun team è coadiuvato dagli operatori della propria sala operativa. La coordinazione tra i team di differenti organizzazioni è realizzata tramite una collaborazione a livello inter-organizzativo

[1] http://www.workpad-project.eu/

tra le sale di controllo dei team coinvolti nello scenario di emergenza. WORKPAD prevede che ogni componente della squadra di soccorso sia equipaggiato con un PDA munito di tecnologie di comunicazione wireless. A supporto di tale organizzazione, il progetto WORKPAD, fornisce una precisa descrizione dell'architettura del sistema complessivo (Figura 1). Tale architettura mette in evidenza la presenza di due livelli organizzativi, chiamati rispettivamente Front-End e Back-End.

Il Front-End rappresenta la rete dei membri delle squadre di soccorso che agiscono direttamente sul luogo ove si è verificata l'emergenza (ad esempio, vigili del fuoco, operatori della protezione civile, ecc.). Tale livello dell'architettura è organizzato secondo il paradigma peer-to-peer: ciascun partecipante può richiedere e/o fornire servizi agli altri partecipanti. La rete di front-end prevede l'esistenza di due tipologie di operatori: il leader della squadra e il generico membro. Il leader dispone di un'applicazione software installata sul proprio dispositivo che permette l'orchestrazione delle attività della missione, rivolte a risolvere l'emergenza. I generici membri dispongono invece di una applicazione che permette loro di offrire servizi e funzionalità debitamente orchestrate dal leader.

Il Back-End rappresenta una rete costituita principalmente da calcolatori tradizionali, il cui scopo principale è quello di coordinare le operazioni svolte dalle squadre di emergenza, fornendo loro obiettivi, dati sensibili e contenuti (ad esempio, mappe interattive). Anche tale livello dell'architettura è organizzato secondo il paradigma peer-to-peer, cioè ogni sistema è in grado di agire sia come fornitore che come consumatore di dati. Ciascun sistema facente parte della rete appartiene ad una specifica organizzazione coinvolta nell'emergenza. Maggiori dettagli riguardanti l'organizzazione dell'architettura del progetto WORKPAD sono specificati nel Capitolo 2.

Il lavoro di tesi svolto dal candidato interviene nell'ambito del progetto WORKPAD, ed ha come oggetto l'ideazione e lo sviluppo di un'innovativa metodologia centrata sugli utenti ed effettivamente utilizzata per coprire tutte le fasi del ciclo di vita del progetto, con l'obiettivo di giungere alla realizzazione di un prototipo finale funzionante del sistema.

Lo sviluppo di un sistema che opera in un contesto critico come quello della gestione delle emergenze richiede che sia garantita una particolare attenzione agli utenti finali del sistema stesso. Infatti, negli scenari di emergenza, diverse squadre di operatori appartenenti a differenti organizzazioni necessitano di interagire l'un l'altra per raggiungere un obiettivo comune. L'eventuale assenza di meccanismi ben precisi che regolino le interazioni in tale contesto potrebbe comportare un peggioramento di una situazione già abbastanza critica, rischiando di mettere in gioco ulteriori vite umane puttosto che salvarle. Ciò significa che l'ideazione e lo sviluppo di una metodologia formale che guidi il sistema interattivo lungo il suo cammino nel ciclo di vita gioca un ruolo fondamentale per ottenere un sistema di successo.

Nell'ambito del Progetto WORKPAD si è deciso di sviluppare una metodologia che enfatizza i punti di vista ed i bisogni degli utenti finali, cercando di adattare il sistema alle loro necessità e non viceversa. Per raggiungere tale scopo, dopo

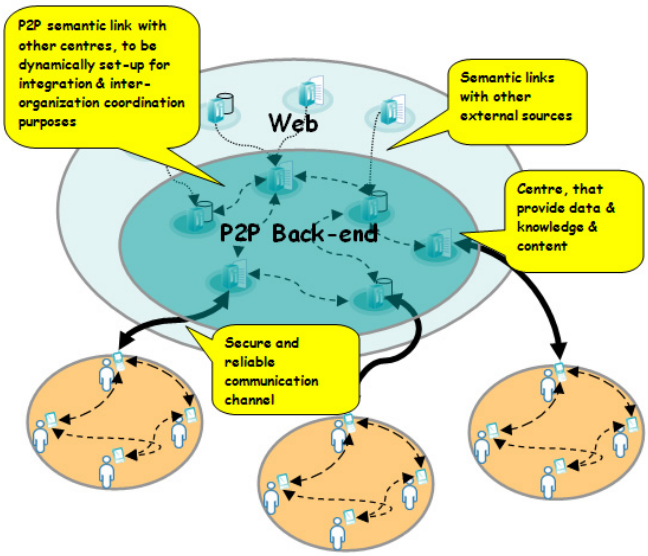


Figura 1: Architettura del Progetto WORKPAD

un'attenta analisi delle tecniche di progettazione a disposizione (Sezioni 3.1 e 3.2), si è stabilito di sfruttare principi User-Centered opportunamente adattati ad un ciclo di vita del sistema iterativo ed incrementale (Figura 2). Le tecniche User-Centered comportano un'interazione continua con gli utenti finali, non solo per comprendere appieno il contesto in cui operano, ma anche per capire che tipologie di informazioni sono per loro rilevanti e le modalità di interazione con gli altri attori del sistema. Come specificato in [3], più il progettista sarà in grado di entrare nella mente dell'utente, maggiore sarà la soddisfazione dell'utente nell'interagire con il sistema.

La metodologia User-Centered sviluppata, che copre gli interi 3 anni di vita del progetto, prevede un'iterazione continua di 3 macro-fasi principali, ciascuna delle quali ha richiesto lo sviluppo di un numero ingente di artefatti progettuali ed implementativi. Tali fasi possono essere sintetizzate come segue :

- **Raccolta ed Analisi dei Requisiti.** La raccolta dei requisiti del progetto WORKPAD è stata effettuata seguendo un duplice approccio e sfruttando le sopracitate tecniche User-Centered. Da un lato, dopo aver compreso come si mette in moto la macchina organizzativa in caso di disastro a livello nazionale (Sezione 2.1) e a livello Europeo (Sezione 3.3), sono stati studiati altri progetti Europei riguardanti la gestione delle emergenze con obiettivi paragonabili a quelli del progetto WORKPAD (Sezione 3.4). Dall'altro lato, è stato condotto un accurato caso di studio nella Regione Calabria, attraverso un attivo coinvolgimento di operatori della Protezione Civile e tecnici operanti nel contesto della gestione delle emergenze. Tale approccio ha previsto il progetto e lo sviluppo di un certo numero di artefatti (Figura 3), la cui

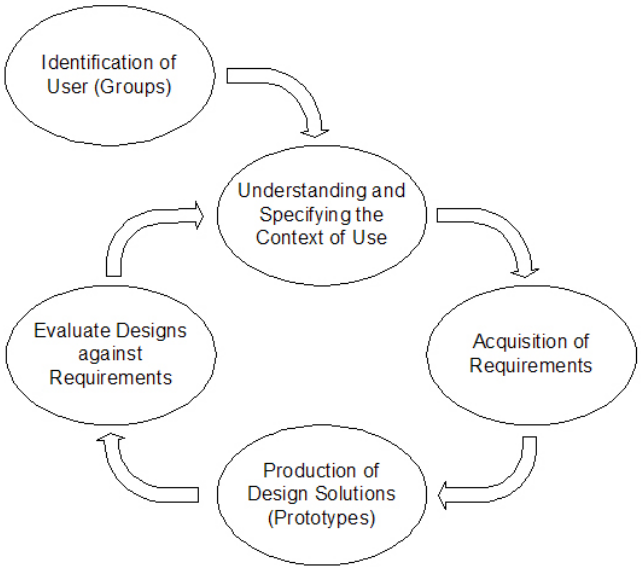


Figura 2: Il Ciclo di Vita Progetto WORKPAD

applicazione ha prodotto risultati fondamentali sfruttati per inferire requisiti validi e raffinarli iterativamente.

Nella primissima fase, attraverso l'utilizzo di interviste personalizzate e questionari (Sezioni 5.1 e 5.2), sono stati identificati gli utenti principali del sistema, cercando di catturarne bisogni ed aspetattive rispetto al contesto studiato. Successivamente, sulla base delle informazioni raccolte, sono stati definiti due macro-scenari di emergenza (Sezione 5.3), che elencano i compiti da eseguire per intervenire rispettivamente dopo un terremoto e dopo un'inondazione. La definizione degli scenari ha rappresentato il punto di partenza inequivocabile per tutte le analisi effettuate successivamente. Il primo risultato degli scenari sono state le cosiddette Storyboards, ovvero situazioni specifiche derivabili da ciascuno scenario. Per ogni storyboard, è stata progettata una Task Analysis, con lo scopo di capire con precisione i passi atomici effettuati da un utente nell'affrontare una situazione reale d'emergenza (Sezione 5.4). Il livello di conoscenza ottenuta ha poi permesso di derivare i requisiti utente (Sezione 5.5), accuratamente descritti attraverso casi d'uso UML (Sezione 5.6). A partire dai Requisiti Utente e dallo stato dell'arte attuale della tecnologia, l'ultimo passo ha permesso di derivare i Requisiti di Sistema (Sezione 5.6). L'Analisi dei Requisiti ha subìto 3 iterazioni distribuite nei 3 anni di vita del progetto, permettendone una raffinazione sempre maggiore.

- **Progettazione ed Implementazione dei Prototipi.** Sulla base dei Requisiti raccolti, i vari partner di progetto hanno cominciato a sviluppare i prototipi

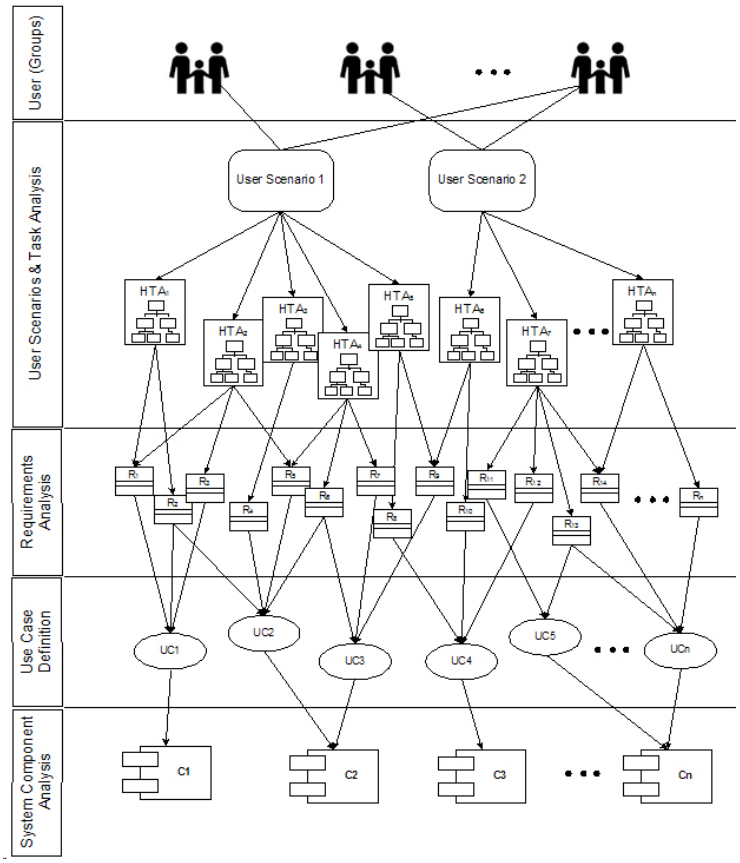


Figura 3: Metodologia per la Raccolta dei Requisiti

dei loro componenti. La Sezione 7.1 mostra l'implementazione e la realizzazione del mock-up del componente preso in considerazione in questa Tesi, cioè il Process Management System (PMS), mettendone in evidenza le funzionalità ed i servizi che esso propone (Figura 4). In particolare, in questo documento si dà una certa enfasi rispetto allo sviluppo dell'interfaccia grafica che, per l'utente finale del progetto WORKPAD, rappresenta senza ombra di dubbio l'aspetto più delicato. Infatti, si ipotizza che un'operatore che intervenga in uno scenario d'emergenza debba interagire con il suo PDA nella maniera più veloce ed efficace possibile. Un sistema con un'interfaccia grafica poco usabile si rivelerebbe controproducente, con il forte rischio di compromettere l'intera operazione volta a gestire l'emergenza. I mock-up e i prototipi dei componenti sono stati raffinati in più iterazioni successive, tenendo in considerazione i requisiti (utente e di sistema) e i feedback raccolti attraverso i test d'usabilità.

- **Valutazione dei Prototipi rispetto ai Requisiti.** Per assicurare che lo sviluppo del sistema soddisfacesse bisogni ed aspettative degli utenti finali, l'ul-

Figura 4: Screenshot del Mock-Up del PMS

timo passo dell'approccio seguito ha comportato la progettazione e l'implementazione di un certo numero di test d'usabilità (Figura 5), necessari per un effettiva valutazione dei prototipi sviluppati rispetto ad i requisiti ottenuti. Tali tecniche di valutazione (qualitativa e quantitativa) hanno permesso di migliorare incrementalmente il prototipo dei vari componenti del sistema, fino a giungere ad una versione finale che soddisfaccesse i requisiti d'usabilità imposti dalla metodologia.

Il primo test d'usabilità è stato realizzato nell'istante in cui i primissimi mock-up dei componenti sono stati rilasciati. Si è trattato di un questionario on-line (Sezione 6.1), in cui gli utenti hanno potuto fornire una valutazione riguardo l'aspetto grafico dei mock-up e la disposizione visiva delle funzionalità sullo schermo del PDA. Il secondo test d'usabilità è stato realizzato con una versione più evoluta dei prototipi iniziali, fornendo agli utenti anche la possibilità di testare alcune funzionalità proposte da ciascun componente. La tecnica usata è stata quella della valutazione cooperativa, che consiste nel far eseguire agli utenti alcuni task e valutarli successivamente insieme a personale tecnico del progetto (Sezioni 6.2 e 6.3). Tale test, i cui risultati sono stati prettamente qualitativi, è stato effettuato con 4 utenti differenti ed è stato videoregistrato. Al termine del test, a ciascun utente è stato chiesto di riguardare il video contenente la propria interazione col sistema e commentarlo, mentre il moderatore prendeva nota dei commenti e chiedeva alcune delucidazioni circa alcuni aspetti dell'interazione. L'insieme dei test d'usabilità è stato completato da alcuni ulteriori test effettuati con utenti non appartenenti al contesto della gestione delle emergenze (Sezione 6.4), con

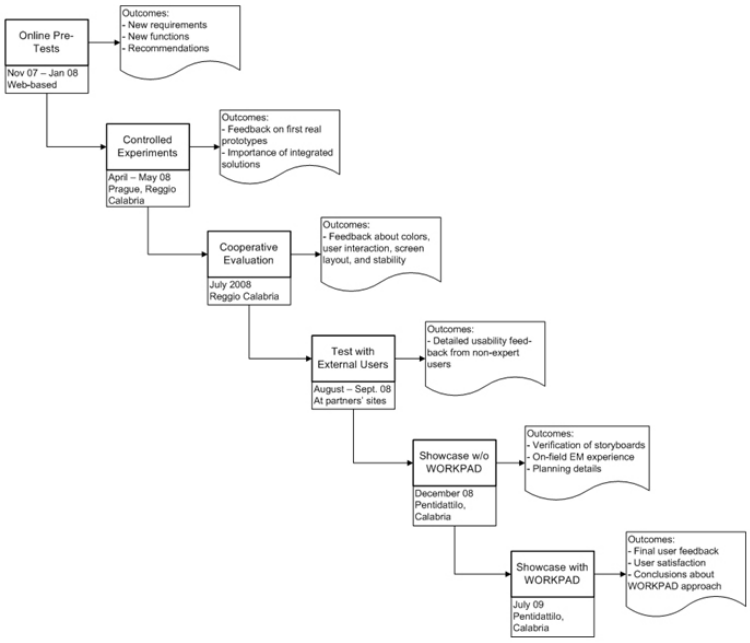


Figura 5: Test d'Usabilità effettuati in WORKPAD

l'obiettivo di fornire un punto di vista completamente differente rispetto ai precedenti. I feedback ottenuti dai suddetti test si sono rilevati estremamente utili ai fini del miglioramento dei prototipi.

A corredo dei test d'usabilità descritti, è stata inoltre organizzata, in collaborazione con la Protezione Civile Calabria, una prima simulazione in un contesto reale d'emergenza senza l'ausilio del sistema WORKPAD (Sezione 6.5). Agli operatori sono stati fatti eseguire alcuni compiti, dando loro indicazione di agire seguendo i piani di emergenza propri della loro organizzazione e le tecnologie a loro disposizione. L'obiettivo di tale simulazione è stato di comprendere in che modo gli operatori affrontavano un'emergenza reale. L'analisi delle modalità di intervento è stata molto utile per la realizzazione dello Showcase finale da effettuarsi con il sistema WORKPAD funzionante.

Con il termine Showcase si intende l'implementazione di un concreto scenario d'emergenza in un contesto reale. Lo Showcase finale del progetto WORKPAD è stato realizzato in Calabria, nella cittadina di Pentidattilo, famosa per un terremoto che la spazzò via nel 1963. La scelta di tale locazione è stata fatta con l'idea di coinvolgere il più possibile psicologicamente gli operatori nel contesto dell'emergenza. Lo ShowCase finale ha permesso di valutare il funzionamento dell'intero sistema WORKPAD, costituito da numerosi componenti software (Sezione 6.6). Ogni utente, ricevuto il proprio PDA con il sistema pre-installato al suo interno, ha effettuato alcuni

compiti delineati da storyboards progettate per l'occasione e simili a quelle sviluppate per la simulazione senza l'ausilio del sistema, il tutto a scopi comparativi. Le attività degli utenti sono state videoregistrate per essere poi successivamente valutate e trasformate in utili feedback per gli sviluppatori.

Il contributo fornito da questa Tesi non si limita all'ideazione della metodologia e allo sviluppo degli artefatti progettuali ed implementativi del progetto WORKPAD (nonostante questo sia il principale argomento trattato), ma comprende anche l'implementazione diretta di un componente software, il Process Management System (PMS). Un PMS è un sistema che permette di definire, creare e gestire l'esecuzione di processi attraverso l'utilizzo di software in esecuzione all'interno di uno o più motori di workflow. Un workflow costituisce quindi un flusso di lavoro composto da un insieme di attività correlate tra loro attraverso diverse tipologie di relazioni.

Nel contesto del progetto WORKPAD, i piani di intervento messi in atto dalle diverse organizzazioni nella gestione di complessi scenari d'emergenza possono essere considerati come processi di business che definiscono le attività da eseguire per il raggiungimento di un certo scopo. Il modello di workflow consente di fornire rappresentazioni dettagliate di tali attività, specificando la struttura dei task da eseguire, i requisiti necessari per l'esecuzione, l'ordine di esecuzione, il flusso di dati coinvolti e le sincronizzazioni tra le diverse attività.

In questa Tesi viene proposto il prototipo finale del PMS basato su situation calculus e sviluppato attraverso IndiGolog, un estensione del linguaggio Golog. Tale PMS non solo può essere utilizzato in classici scenari di business, ma si rivela molto utile anche in contesti altamente dinamici e incerti, come quello della gestione delle emergenze. Maggiori dettagli circa l'implementazione del PMS sono forniti alla Sezione 7.2.

Per concludere, si vuole sottolineare che i tutti i risultati ottenuti e descritti in questa Tesi hanno contribuito al processo di disseminazione del progetto WORKPAD, grazie ad una serie di pubblicazioni scientifiche che ne hanno riconosciuto l'alto grado di innovatività (per maggiori dettagli, vedere il Capitolo 8).

Contents

Chapter 1

Introduction

The widespread availability of network-enabled handheld devices (e.g., PDAs with WiFi/UMTS capabilities, TETRA smart terminals, etc.) has made the development of pervasive computing environments an emerging reality, very suitable for managing emergency/disaster situations.

Disaster is a broad term; it can be defined as a serious disruption of the functioning of a community or a society causing widespread human, material, economic or environmental losses which exceed the ability of the affected community or society to cope using its own resources [5].

Emergency management means all the coordinated activities carried out to prepare, support and rebuild society when natural or human-made disasters occur. Such activities, as suggested by [2], can be grouped into five phases (see Figure 1.1) and structured by time and function for all types of disasters.

The preventive measures are divided into planning, mitigation and preparedness activities. During the *planning* phase it is necessary to analyze and document the possibility of an emergency event or a disaster and the potential consequences or impacts on life, property and environment. The results of this phase are essential for the next preventive phases. *Mitigation* activities eliminate or reduce the probability of a disaster. It includes long-term activities designed to reduce the effects of unavoidable disasters. In the preparedness phase governments, organizations and individuals, develop plans to save lives and minimize disaster damage. *Preparedness* measures seek to enhance disaster response operations. When an emergency happens, the *response* phase is designed to provide emergency assistance for victims and to stabilize the situation reducing the probability of secondary damage. The *recovery* activities aim at returning vital life-support systems to a minimum operating standard.

These last two phases are actually the most critical, and are the focus of the European project WORKPAD[1], whose purpose is to provide a software and communication infrastructure supporting operators facing an emergency.

The development of a system working in such a critical environment requires a

[1] http://www.workpad-project.eu/

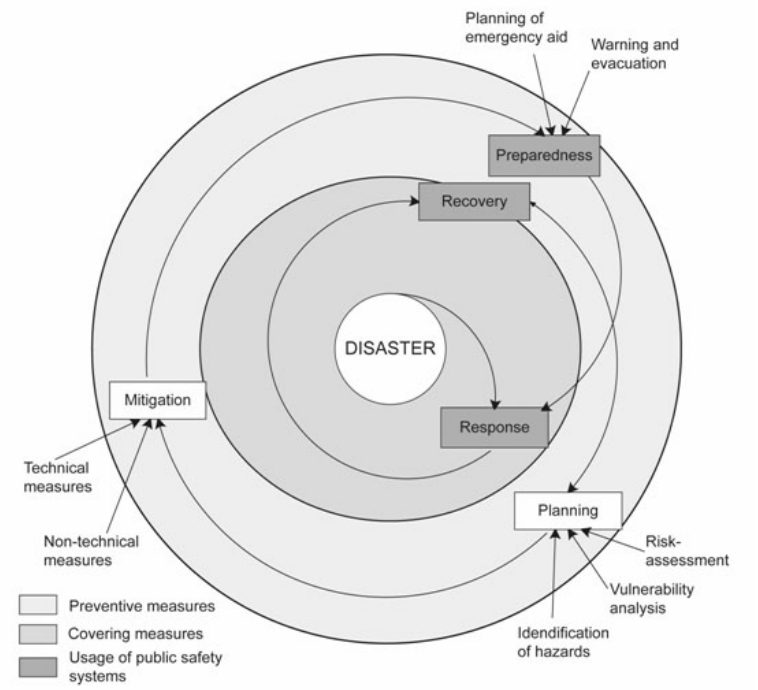


Figure 1.1: The life cycle of disaster management

particular attention for many aspects as a small mistake can put many human lives in danger. In emergency scenarios, it is required that different teams belonging to different organizations collaborate each other to reach a common goal; the lacks of the basic interaction principles can be dangerous as it could increases the level of disaster or can make the efforts ineffective in such scenarios. This means that a formal methodology and well-designed principles, for designing and developing an interactive system for supporting emergency operators, plays a critical role in the success of the resulting system.

This thesis aims to show the methodology used for designing the WORKPAD interactive system. The process of designing interactive systems does not involve only the interfaces or the immediate interaction; rather, it relates to the complete environment (e.g. the technology, working environment, stakeholders, goals, etc.) surrounding that system; and this process produces a set of resulting artefacts (e.g. documentation, manuals, tutorial, etc.) along with the working system to make the interaction more effective. In order to devise a successful information, communication, and media technology (ICMT) architecture for emergency management, it is decided to employ User-Centered techniques from human-computer interaction paradigms [3]. User-Centered design relies on continuous interaction with end users to understand how organizations are arranged during disasters, what information is critical, and how teams exchange this information among themselves and with their operational centers. As specified in [3], the more designers will be

able to go into the "mind" of actors, the more the system will match user needs and be appreciated by users.

In the WORKPAD project, twofold (*bottom-up* and *top-down*) high-level approaches with various human-computer interaction (HCI) techniques were selected for taking the requirements and to design the system. The *top-down* approach is used to get information regarding the related works, while the *bottom-up* approach is used to get requirements from the practical work carried out in the field. It has been also used the experience knowledge of users and technical persons working in the emergency or disaster scenarios to get more User-Centered focus. The work done according to the selected approach is as follow:

- **Bottom-up approach** A concrete case study of emergency management in Calabria (an Italian region) has been conducted. Potential users have been intensively involved in this project phase according to the HCI techniques delineated in International ISO standard 13407 [1]

- **Top-down approach** On the one hand, it has been investigated European legislation, recommendations and initiatives with respect to emergency management, and on the other hand, related European research projects have been examined regarding the requirements analysis methods adopted, the concrete outcomes and their validity for the WORKPAD project

To summarize, this thesis wants to show how to conduct with success a European Project which design is mainly based on User-Centered design techniques. Moreover, it will also give details about the concrete implementation of one of the main components the project; the Process Management System (PMS).

The thesis is structured as follow (for a rational reading, make reference to Figure 1.2) :

- **Chapter 2** after providing a description of the Emergency Management in Italy (giving details about the Calabrian case), provides an overview of the WORKPAD project, describing its motivations, basic requirements and goals. It also introduces the system's architecture, with a focus on Front-End and Back-End architecture. The Chapter terminates presenting the roadmap of the activities conducted by this Thesis within the WORKPAD life cycle.

- **Chapter 3** details the State of the Art about the software designing principles (Plain Driven Methodologies, Human Computer Interaction techniques) and the "knowledge base" (status of emergency management in Europe, analysis of other IST projects) needed to understand how to design the project.

- **Chapter 4** describes the adopted methodology and the approaches that were used to make the system more interactive and more efficient in emergency scenarios.

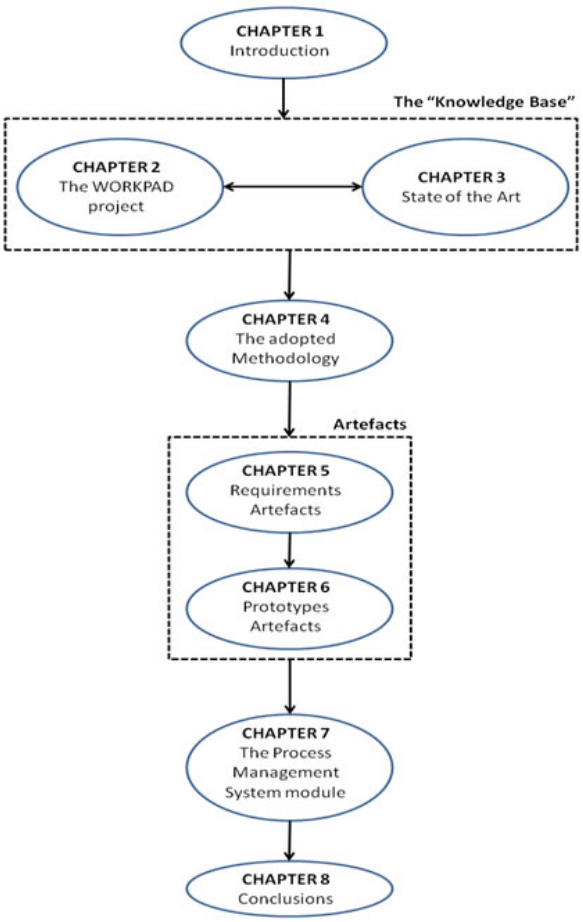


Figure 1.2: Structure of the thesis

- **Chapters 5-6** explain in detail the activities conducted during designing and developing the WORKPAD project, showing the resulting outcomes and artefacts (distinguished by requirements artefacts and prototype artefacts).
- **Chapter 7** shows, at the beginning, the design and the implementation of the mock-up realized for the Proces Management System, with a particular focus on the Graphical User Interface. Then, after a brief introduction about Process Management, it describes the framework and the implementation of the Process Management System realized in IndiGolog for the project.
- **Chapter 8** gives some conclusions, underlining the dissemination effort carried out by this Thesis, by a list of all the papers published on the basis of the results obtained.

Chapter 2

The WORKPAD project

The European research project WORKPAD aims at designing and developing an innovative software infrastructure (software, models, services, etc.) for supporting collaborative work of human operators in emergency/disaster scenarios. In such scenarios, different teams, belonging to different organizations, need to collaborate with one other to reach a common goal; each team member is equipped with handheld devices (PDAs) and communication technologies, and should carry on specific tasks. Referring to the cycle of disaster management shown in Figure 1.1, WORKPAD focuses on the response and short-term recovery phases. The response phase begins shortly after the emergency has happened. Response activities are designed to provide emergency assistance for victims. The short-term recovery phase follows the response phase and aims at returning vital life-support systems to a minimum operating standard.

To devise successful information, communication, and media technology (ICMT) architectures for emergency management, WORKPAD employs User-Centered techniques from human-computer interaction paradigms. User-Centered design relies on continuous interaction with end users to understand how organizations are arranged during disasters, what information is critical, and how teams exchange this information among themselves and with their operational centers. The application of user-centered techniques, in collaboration with the Civil Protection of Calabria, Italy, involved interviewing officers and generic actors from the organizations most critical to emergency management in that region.

From the understanding of how Civil Protection works in Italy during an emergency, two typologies of users - Back-End and Front-End users - have been identified. Front-End users are all the operators acting directly on the field during disasters (ranging from firemen to voluntary associations). Back-End users are all the operators who manage the situation from control rooms, by providing goals/instructions/information to Front-End operators. Typically, at Front-End, each team has a leader that takes decisions; inside the team, communications usually are performed by transceivers and mobile phones. Moreover it is not possible to communicate directly with members of other organizations: all the inter-team co-

ordination is at the Back-End level, where possibly officers have got more data and information which are helpful to make choices.

This Chapter aims to give an overview of the architecture designed for the WORKPAD project, investigating about the context in which the system should operate.

- Section 2.1 furnishes a precise description about how Italian Civil Protection works during emergency management.
- Section 2.2 gives an overall idea of the WORKPAD project, emphasizing the high level architecture devised for the project and showing a possible scenario in which the system could operate.
- Section 2.3 gives an overview about the components of the Front-End subsystem.
- Section 2.4 gives an overview about the components of the Back-End subsystem.
- Section 2.5 shows how the link between Front-End and Back-End could be performed.
- Section 2.6 presents the roadmap of the activities conducted by this Thesis within the WORKPAD life cycle.

2.1 The Emergency Management in Italy

The Italian expression "Protezione Civile" [Civil Protection] is to be understood as the series of actions and activities put forward in order to protect human lives, goods, settlements and the environment from damages or from the danger of damages deriving from natural calamities, catastrophes and other disastrous events. The response to disasters is regulated in Italy by the Law 225 (24 February 1992): it organizes the Civil Protection as a national service, politically coordinated by the premier. It consists of central and local authorities, and includes public corps and private institutions existing in the national territory. Specifically, the "Augustus method", strictly linked to the Law 225, describes how to plan during disasters and clearly delineates a method for response activities. It gives guidelines and general procedures to effectively coordinate Civil Protection activities. To cope unpredictability in disaster occurrence, adaptive strategies have been devised by the Italian Civil Protection [28], based on the observation that - as the Emperor Augustus used to say - "the value of a plan decreases as the complexity of the situation grows". Such strategies are meant to provide a suitable approach to the problem of coordinating complex operations according to flexible plans in disaster contexts. The Civil Protection service consists of central and peripheral administrations, Regions, Provinces, Municipalities, national and territorial public agencies, and any

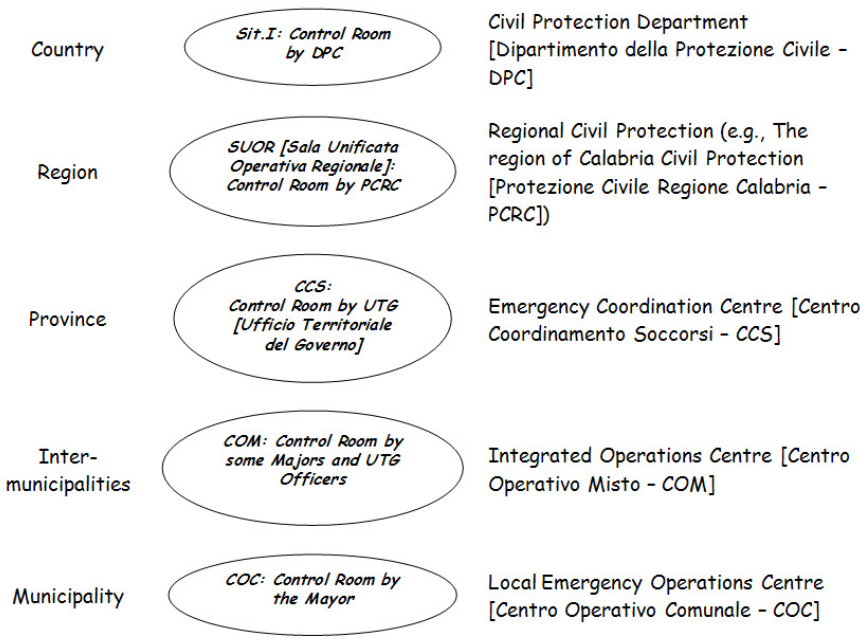


Figure 2.1: The five levels in Italian Emergency Management

other public and private institution and organization present on the national territory. The President of the Council of Ministers provides for the coordination of the national service and the promotion of emergency management activities through the Civil Protection Department. Figure 2.1 shows the five hierarchical levels of the Civil Protection service, together with relevant offices/centres.

The service is set up on the principle of subsidiarity: in each Municipality, the person officially responsible for Civil Protection is the Mayor, who organizes municipal resources according to pre-established plans made to cope with specific risks in his territory. When a disastrous event occurs, the service is able, in a very short time, to define the event's significance and assess whether local resources are sufficient to face up. If needed, the support of other Municipalities, Province(s) and Region(s) is guaranteed and coordinated by the Prefects. In the most serious situations, a national integration will take place: forces available are united with any other staff and equipment necessary to effectively meet the needs. In such cases of national emergency, the leading role is no more assigned to Prefects but to the Civil Protection Department, and the President of the Council of Ministers assumes the political responsibility.

An overview of this structure in the specific case of an emergency in the Region of Calabria is shown in Figure 2.2. At regional level, the Civil Protection in Calabria has the availability of an operational control hall, called ***SOUR*** - *Regional Unified Room for Operations* [*Sala Operativa Unificata Regionale*]. Here, two persons permanently (i.e., 24 hours/day) monitor the situation. Usually, SOURs

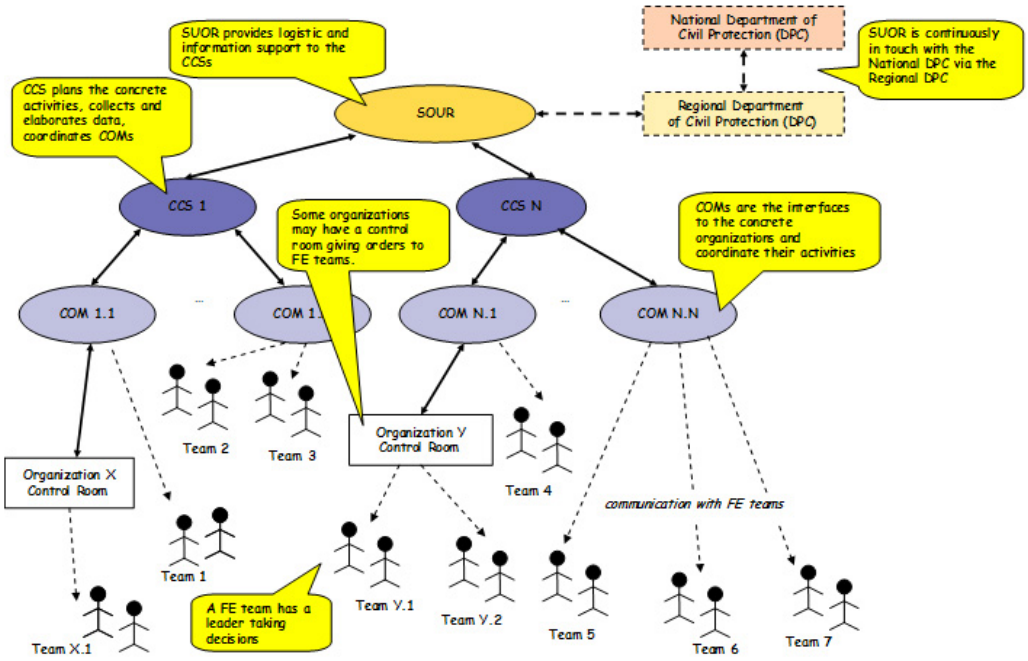


Figure 2.2: Overview of the structure of Italian Emergency Management hierarchy

are equipped with basic infrastructures to get information and to communicate: fax server systems, radio communication systems, hydrometeorological monitoring control systems, etc. The SOUR's activities are focused to control possible alerts, e.g., from the hydrometeorological system, to receive, to check and to manage any kind of event and to issue alert messages.

SOUR is continually in touch with the National Civil Protection Department and, when a calamitous event happens, with the prefectures in the affected areas, in order to furnish them a logistic and informative support. When a disaster breaks out, the actual coordination of required activities is in charge of the Province (Calabria consists of five provinces: Catanzaro, Cosenza, Crotone, Reggio Calabria and Vibo Valentia). These tasks include the activation of the ***CCS*** - *Centre for Coordination of Aids* [*Centro Coordinamento Soccorsi*], which represents the strategic-operational top line of the Civil Protection at this level (one CCS exists in each Province). CCS is coordinated by the Prefect and is composed by a fixed number of officers of the organizations (Police, Fire Brigades, etc.) which need to be always involved in emergency management (according to the Augustus method). Moreover, according to the specific disaster, CCS can be integrated by officers of other organizations, whose competences can be useful to face the specific situation; e.g., ANAS [National Agency for the Road Network] is involved in scenarios where roads have to be restored. Main CCS tasks are inspections, the collection

and elaboration of data and information concerning the evolving situation, the continuous connection with SOUR and the coordination of all activities performed by COMs.

A ***COM*** - *Mixed Operational Center* [*Centro Operativo Misto*] is an operative decentralized structure depending on the CCS. Each Province can have more COMs arranged in its area (so, it means an inter-municipal level), e.g., the Province of Reggio Calabria has 19 COMs (see also Figure 2.3). The constitution of a COM comes from the necessity to organize inspections as quickly as possible on the territory affected by a calamitous event. A COM is closer to the disaster: it acknowledges immediately the different local demands and organizes the work to be carried out. COMs should return any result to CCS, as CCS has to get a complete and updated scenario and can coordinate the work of several COMs. The Director of a COM is a Prefect's officer. The participants are Municipalities and operative structures representatives (Police, Fire Brigades, Red Cross, Voluntary Associations etc.). The specific tasks of the COM can be summarized as :

- coordination of all rescue operations;
- food and water supply;
- other interventions to handle an emergency.

The lowest level of emergency management is at municipal level; here the Mayor is the first line of defence. Mayors coordinates emergency by using ***COC*** - *Local Emergency Operations Centre* [*Centro Operativo Comunale*]. He has to receive, through the COC, all damages reports and help requests. As the Mayor is not able to operate directly, he himself pass all information to COM. Therefore, the importance of COC take place only in minor emergencies.

Nowadays, communications between operating structures (Front-End teams, COMs, CCSs) take place by telephone and/or fax. This is a first critical point, as it allows only personal one-to-one communication without using any analytical or graphical tool, e.g., *GIS*s (*Geographical Information Systems*). The organizations (Fire Brigades, Police, etc.) in COMs communicate with their respective control rooms, through radio frequencies or, if unavailable, by telephone and fax. For some organizations, control rooms might not exist, and therefore COMs communicate directly with Front-End teams. Usually, each control room has a computer system where data and information concerning the disaster are collected and stored. Nowadays this information is not directly shared with any other organization. At Front-End, usually each team has a leader that takes decisions; inside a team, communications typically are performed by transceivers and mobile phones. Moreover it is not possible to communicate directly with members of other organizations: all the inter-team coordination is at the Back-End level, where possibly officers have got more data and information which are helpful to make choices.

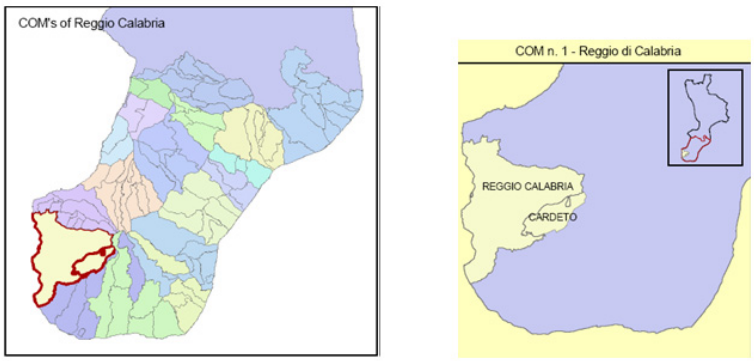


Figure 2.3: Example of COMs in the Region of Calabria

2.2 The System Architecture

The high-level design of the WORKPAD architecture has been established based on

- the results of the requirements elicitation process (see Chapter 5 for details);
- the analysis of the potential users - Calabrian Civil Protection - performed in section 2.1;
- the theoretical examination of related projects which contained EU regulations in order to derive valuable requirements for the WORKPAD system (see Sections 3.3, 3.4 for details);
- the objectives and focus of the WORKPAD project.

WORKPAD architecture is based on a 2-level peer-to-peer (P2P) paradigm: the first level is for the Back-End, and the latter one is for the Front-End. Figure 2.4 shows the WORKPAD's architecture. The WORKPAD Front-End level is made up of several teams. The members of a single team belong to the same organization and are equipped with mobile devices (e.g., PDAs). Team members establish a P2P Mobile Ad-hoc NETwork (MANET) for coordination and intra-team communication. A MANET is a P2P network of mobile nodes capable to communicate with each other without an underlying infrastructure. Nodes can communicate with their neighbors (i.e., nodes in radio-range) directly by wireless links. Non-neighbor nodes can communicate by using other intermediate nodes as relays which forward packets toward destinations. Every node has to keep and update routing tables to know usable paths to forward data packets to destinations.

The lack of a fixed infrastructure makes this kind of network suitable in emergency management scenarios where it is needed to quickly deploy a network but the presence of access points is not guaranteed. As well, since the available bandwidth (roughly 11 Mbps) is enough, MANETs can guarantee a good QoS level.

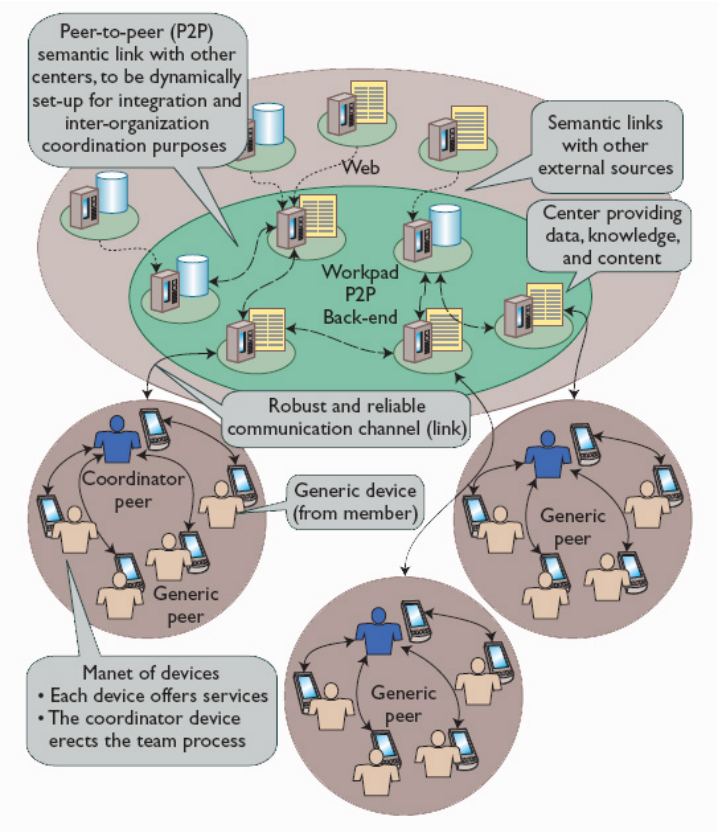


Figure 2.4: WORKPAD high-level architecture

The WORKPAD Back-End is a peer-to-peer overlay network including back-office systems (e.g. services, data bases) of the operating organizations. By entering WORKPAD network, Back-End peers can easily integrate their data, content and knowledge which are gathered and used for emergency managements. Front-End operators access the Back-End network through their own back-office systems where they can get or set information which is relevant to the situation they are facing or the action they are taking. Thanks to the integration layer, such information is not necessarily contained in single systems, but is potentially spread over the network, and is delivered, collected, reconciled on demand.

Not all devices can be equipped with costly and reliable technologies for communicating between the Front-End teams and the Back-End. Instead, in WORKPAD, all devices can communicate with each other through the manet, while a few of them (the team leader's device and a few backups) act as gateways to the Back-End. Then it's possible to equip the gateway devices with various technologies (the satellite, the Universal Mobile Telecommunications System [UMTS] and Terrestrial Trunked Radio [Tetra]). Devices can switch among these technologies depending on their availability.

In order to give to the reader a first idea of how the system should work in a real scenario, the subsection 2.2.1 describes a first concrete example.

2.2.1 An operative scenario

Before further detailing the key components of the Front-End and Back-End, this subsection briefly describes how the system should work. As an example, consider the following scenario of disaster recovery. After an earthquake (or a hurricane), a team (e.g., belonging to the Civil Protection Department), equipped with mobile devices (laptops and PDAs), is sent to the disaster area to evaluate the state of specific sites. Their goal is to document the damage directly on a situation map, and to schedule following activities (e.g., reconstruction jobs). Before this process starts, the team leader has stored all area details, including a site map, a list of the most important objects at the site, and some previous reports and materials. All such details have been provided by the Back-End centre of the Civil Protection Department, which has constructed them by integrating information & knowledge & content stored by many other peer organizations (e.g., the Ministry of Internal Affairs, some basic spatial data provided by different public and private organizations, etc.). It is worth observing that the integration and involvement of a particular peer has been dynamically and adaptively decided on the basis of the specific process, which in turn depends on the given emergency situation. Therefore no pre-existing, defined integration infrastructure exists among the Back-End peers, but it is rather dynamically built on-demand.

On the Front-End, the team constitutes a MANET (Mobile Ad hoc NETwork), in which the team leader's device coordinates the other team members devices by providing appropriate information (for example, maps, important objects, and so on) and assigning activities. MANET are networks of mobile devices that communicate with one another via wireless links without relying on an underlying infrastructure. This distinguishes them from other types of wireless networks - for example, cell networks or infrastructure based wireless networks. To achieve communication in a MANET, each device acts as an endpoint and as a router forwarding messages to devices within radio range. MANET are a sound alternative to infrastructure-based networks whenever an infrastructure is no longer available, or can't be used, as in emergency scenarios [24].

As an example, consider what happened recently during the Katrina's emergency in New Orleans, USA: among the different communication infrastructures, only the satellite-based one survived, but it was not possible to use it for all communications of the teams working in the area, with the known dramatic consequences. Conversely, teams equipped with PDAs forming a MANET would be able to communicate each other, and exploit a possible satellite channel available to one of them (e.g., the team leader) for possible coordination with other teams. The team members devices enable them to execute some operations. Such operations, possibly supported by particular hardware (for example, digital cameras, different communication connections, computational power for image processing, main storage,

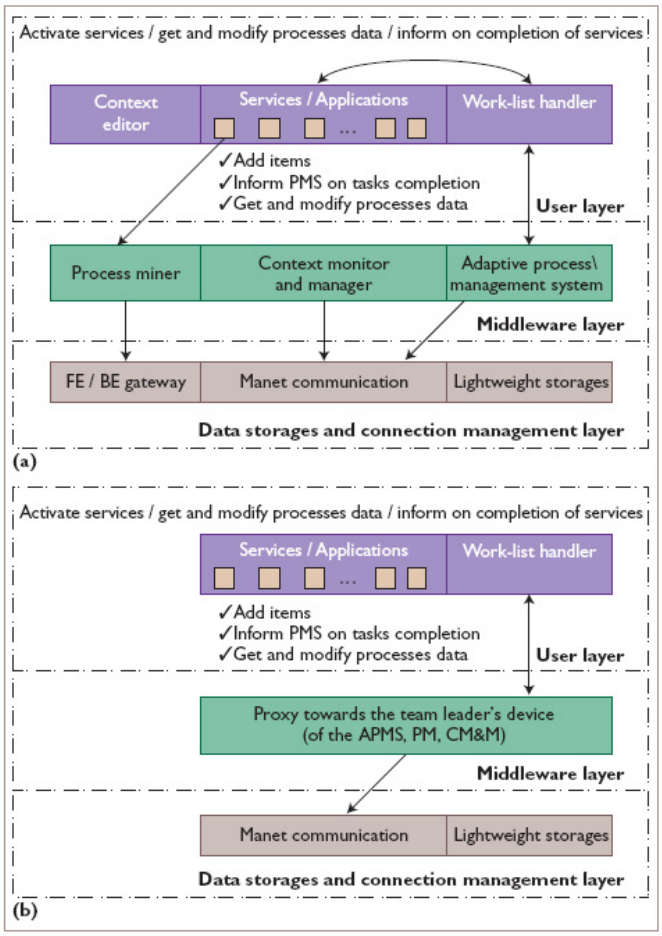


Figure 2.5: Overview of components of the WORKPAD Front-End (a) The team leader's device, and (b) a team member's device

etc.), are offered as software services to be coordinated. Such a coordination is carried out by a specific coordination layer hosted by the team leader device.

2.3 The WORKPAD Front-End

Figure 2.5 shows the conceptual architecture at the WORKPAD Front-End. It consists of three layers named, from the top to the bottom, ***User Layer***, ***Middleware Layer*** and ***Data Storages and Connection Management Layer***. Each layer consists of several components. Although the architecture shows various components at the Front-End, not all components have been deployed in every Front-End devices. Instead, the deployment of these components will be customized, depending on the capability of devices and, probably, the role of the team member who con-

trols the device. Moreover, each component will also be customized for specific devices. For example, the quality of figures captured by an application might vary; however, if a device is equipped with a camera, then the application should be able to utilize the camera.

The **Data-Storage and Connection-Management Layer** includes two modules:

- *the manet communication module*, which implements manet multi-hop communication;
- *the lightweight storage module* for data and knowledge storing (either local or distributed).

Current operating systems don't allow communication among non-neighboring wireless peers, so there is the need of a specific software module that implements one (or more) MANET algorithms. The team leader's device, in addition, holds a further module named *Communication Front-End/Back-end* to handle connections with Back-End.

The core element of the **Front-End Middleware Layer** is the *adaptive Process Management System (aPMS)*. It adaptively controls emergency management processes based on contextual information retrieved by the *context monitor and manager*. This contextual information is associated with devices, networks, team members, activities, and so on. A *process miner* detects workflow patterns, individual member and team social behavior, and possible correlations. The middleware layer at generic nodes is simpler, consisting only of specific modules whose purpose is to interact with the team leader's counterparts.

Concerning the **User Layer**, when the aPMS assigns a task to an actor, it inserts the task into that actor's *worklist handler*(one for each device). Users learn their assigned tasks by querying this list. When they're ready to perform a given task, they pick the corresponding item in the work list together with data needed for its execution. The handler knows which skill (service) is required for its execution, and, according to the required service, the handler runs the corresponding application to provide the service. When the application is closed, modified data returns to the handler with a notification of task completion. The handler forwards the data and termination to the aPMS. In Team Leader's device, the *context editor* component lets users enter additional contextual information that the Front-End middleware couldn't capture. Specific services offered by the devices are deployed based on the corresponding team member's capabilities and skills.

2.4 The WORKPAD Back-End

WORKPAD Front-End networks are connected to specific Back-End systems, which include a Web services platform to allow data exchange and integration. This platform is designed as a P2P network, in which each system (peer) can act as

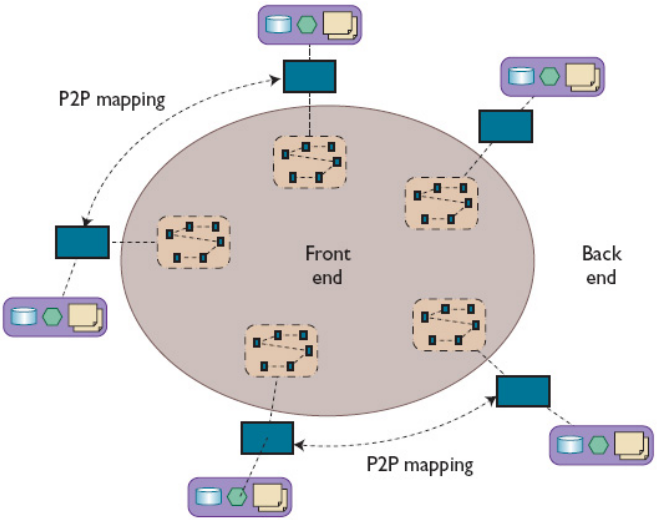


Figure 2.6: Back-End peers form an overlay network

data provider, consumer, and integrator. Typically, a peer may represent one of the organizations (i.e. Civil Protection, Police, Fire Brigades, etc) that are involved in the course of emergencies.

By plugging into the WORKPAD's Back-End network, a back-office system qualifies as a WORKPAD Back-End peer. This peer exports its ontology (that is, a schema reflecting its conceptual model); allows a rapid integration of various data sources, both internal and external(including other peers), through mappings from available sources to the ontology; and can answer conjunctive queries expressed in the alphabet of ontology terms. Front-End services or other Back-End systems can issue these queries. A peer can also receive notifications of relevant updates in other systems in the network. By mapping their ontologies onto one another, peer systems achieve semantic integration without resorting to global ontologies. Obviously, peers can also conform to shared conceptualizations. Moreover, the more peers agree on shared conceptualizations, the fewer mappings between ontologies are needed. The ultimate goal, however, is to relieve peers from having to adopt large and complex shared ontologies, thus facilitating their rapid integration. The integration and involvement of a particular peer is dynamically and adaptively decided on the basis of the specific process to be put into effect. This process, in turn, depends on the given emergency situation. The integration logic will therefore be distributed throughout the entire system, so peers will need no specific integration systems beforehand.

WORKPAD Back-End peers are systems that perform ontology-based knowledge retrieval over a set of local sources and a specific overlay network. Local sources can be relational database management systems, Web services, XML documents, and so on. WORKPAD also gives knowledge administrators a set of tools for managing and controlling peers. These tools provide utilities that let adminis-

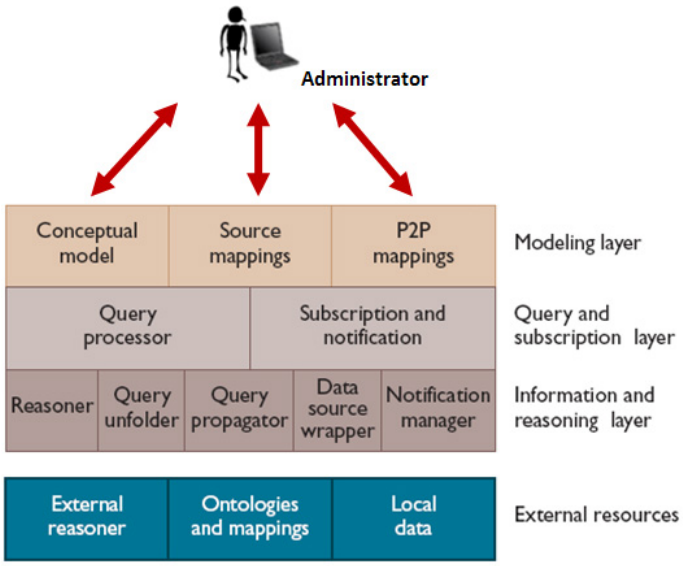


Figure 2.7: Conceptual architecture of a Back-End peer

trators define organizational ontologies and data sources, specify conceptual mappings between ontologies, and so on (see Figure 2.6). The peer functionalities have been organized in three layers: ***Modeling layer***, ***Query and Subscription layer***, and ***Information and Reasoning layer*** (see Figure 2.7).

The **Modeling layer** comprises the functionalities that let knowledge engineers define peer ontologies, map data sources (local data) onto these ontologies, and specify conceptual mappings between locally defined ontologies and those defined by other peers. All of these functionalities are built over a rich client platform based on the Eclipse technology.

The **Query and Subscription layer** manages conjunctive queries (CQs, that is, a restricted form of first-order queries) and subscription requests. It also supports notifications of data and knowledge changes. The *query processor* component is in charge of reformulating the input query in terms of i) concrete queries toward local data sources and ii) CQs toward related peers. Clients (Front-End services or external systems) will be able to subscribe to any conjunctive query so they can be notified when a query's extension changes (that is, when new records are added, deleted, or changed), provided that the client supports callback functionalities compliant with the WS-Notification specification [30]. The *subscription and notification* component will handle these functionalities.

At the **Information and Reasoning layer**, clients' concrete queries are propagated to other peers or executed toward local data. The *query propagator* and the *datasource wrapper*, respectively, perform these two functions. The *notification manager* receives a message each time the local data are modified. This message

triggers the upper layer's subscription and notification component, which consequently notifies all subscribed end points for queries that are impacted by such changes. Any application that changes the local data must promptly send a message to the notification manager. The *query unfolder* reformulates the input query in terms of concrete queries toward local data sources and conjunctive queries toward related peers. The *reasoner* provides the necessary reasoning capabilities for reformulating queries.

Moreover, it exists also a further layer consisting of all the external resources that are relevant to the Back-End peer, named *External resources layer*. These resources consist of the external reasoners, the materialization of the ontologies in suitable databases, and the mappings and local sources (or local data).

Finally, without entering in detail, it is necessary to underline that the WORKPAD Back-End has to support the exchange of geo-information provided in a variety of formats and belonging to different organizations. In doing so, WORKPAD uses GIS services and integrate them into the Back-End.

2.4.1 P2P Information Integration

WORKPAD's Back-End is a flexible data-integration infrastructure that supports P2P networking, in addition to classic mediator-based architectures. In general, data integration involves combining data from different sources and providing the user with a unified and consistent view of that data. Most current data-integration platforms are based on *centralized semantic mediation* - that is, a single system provides a mediated schema (a global view) over distributed data sources and performs sound and complete query answering over them. Semantic mediators take full responsibility for interpreting data sources because they're generally defined within the same epistemic boundaries. So, the system acts as if it's a single database, in which query answering is given standard first-order semantics.

On the other hand, P2P data integration combines autonomous systems and so must answer queries using possible-world semantics. From a single peer's viewpoint, the key issue is how to use information gathered by other peers. One possibility is to adopt a global semantics-based approach. In this case, WORKPAD peers could draw mappings between their concepts and others'.

In general, a query to a P2P networking system must account for the epistemic difference between what the queried system knows (that is, actual data) and what it knows the others know (that is, information collected on the network). In other words, it must distinguish between the knowledge of facts and the knowledge of what is referred. WORKPAD proposes a doxastic approach such that peers don't transfer their knowledge of facts to others. Instead, for a specific peer A, the knowledge of another peer B becomes knowledge of that peer's opinion. Intuitively, whereas an epistemic approach aims to model knowledge of facts with respect to many possible states of affairs, a doxastic approach separates facts from opinions, giving the latter a specific ontological status. Epistemic systems such as Hyper require peers to deal with inconsistencies. For example, contrasting knowledge (such

as Damaged(e) at peer A, and not(Damaged(e)) at peer B), if not repaired, would lead to global uncertainty about the item referred to. For this reason, a doxastic approach seems to be appropriate in situations in which inconsistencies might easily arise and can't be easily repaired. Actually, this approach directly leads to the consequence that knowledge of facts by peer A can't be inconsistent with knowledge of the same facts by peer B, because the knowledge of these two peers is formally separated.

Starting from a doxastic approach, WORKPAD uses modal queries with a special syntax to request beliefs of the queried peer. The peer computes the result set of an epistemic query, appropriately querying directly connected peers, and returns the set to the client. This result set is an aggregation of facts (from its own data sources) and beliefs (from mapping other peers' result sets). Each belief will include information about which peers consider it a fact, thus tracking data provenance. The query issuer will then have the task of managing and composing potential conflicts between different opinions across the network.

2.4.2 Concept Languages and Reasoning

Ontology development and mapping play an important role in WORKPAD. In fact, due to its P2P nature, WORKPAD does not rely on any model established beforehand (except those of geo-referenced data). WORKPAD supports scenarios in which peers provide their own conceptualizations, based on its data sources. WORKPAD Back-End systems will manifest their schemas by using an expressive concept language, possibly by mapping their existing schemas (for example, relational). For data-intensive applications such as WORKPAD applications, the trade-off between these languages' expressive power and the computational complexity of sound and complete reasoning is crucial. Specifically, the most relevant reasoning task for a peer is answering to conjunctive queries over the set of instances maintained in its secondary storage.

Current research often represents ontologies using formal languages from the description logics family. Rich description logics based on current Semantic Web standards (such as OWL) typically suffer from worst-case exponential time reasoning when applied to query-answering tasks. In general, borrowing from database theory, we can evaluate the query-answering problem's complexity with respect to the query's size (query complexity), the data's size (data complexity), or both (combined complexity). When a system must deal with a large amount of data, you should keep data complexity as low as possible. For this reason, ontologies used in WORKPAD uses a particular description logics subfamily of languages, called DL-Lite. DL-Lite is rich enough to capture most basic ontology languages, such as UML class diagrams, while keeping the reasoning com-plexity low. DL-Lite standard reasoning tasks are polynomial in the size of ontology terms (TBox), and query answering is polynomial in the size of individual assertions (ABox). Noticeably, query answering is polynomial (*logspace*) in data complexity - that is, it has the same complexity of traditional database management systems. Moreover, DL-

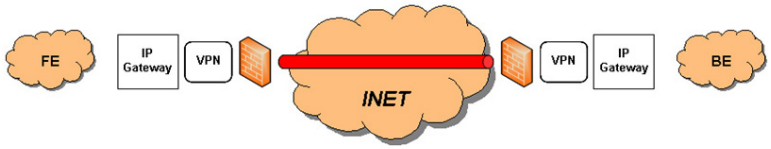


Figure 2.8: The FE/BE link set-up

Lite lets a system separate terminological from individual reasoning, so existing database management systems can perform processes requiring data access using suitable query-rewriting processes.

2.5 Front-End To Back-End Link

Depending on disaster scenarios where it is possible to have limited network infrastructure available or none in the worst case, there can be several suitable solutions for establishing the connectivity between the Front-End and the Back-End. The key issue is that this eventual situation must be "transparent" to the users in the Front-End so that they can rely on different alternatives in the communication with the Back-End. Thus, they can do their job in an appropriate way without worrying about the change of the underlying networks. Moreover, WORKPAD doesn't aim at developing Front-End components that work on various transport network protocols, which are not compatible and easily to be exchanged. Instead, such an issue is solved by using existing solutions, e.g. using bridge or conversion. Therefore, Front-End and Back-End software use a communication based on standards such as UDP/IP or TCP/IP stack. Information has to be transmitted over reliable wireless channels because the data exchanging between Front-End and Back-End is crucial when teams are working, and because a networking black-out (i.e., a Front-End/Back-End link failure) in such phase could be fatal for all the emergency management. In this sense, this kind of communication technologies can be given using either GPRS/UMTS, TETRA, satellite or WiMAX since all of them can rely on an IP network with UDP/TCP as transport protocols.

In order to achieve an integrated infrastructure in which Front-End nodes can have access to the Back-End, a team (a MANET community) is connected to wired networks through wireless links, such as cellular (e.g., UMTS technology), satellite or TETRA solutions acting as routers. Teams are asymmetrical MANETs, where there are some special nodes (e.g., the coordinator device) acting as a gateway to the fixed network. Such nodes are equipped with at least two network interfaces: one connected to the MANET and the other one to the Back-End, and are able to receive and propagate traffic from outside hosts and route the traffic to the destination MANET node. Such devices play a critical role and may become bottlenecks.

Based on the assumption that, in extreme cases, an IP connection is always available to link the Back-End infrastructure to the Front-End, the election of the

most suitable technology used for this communication will depend on the following characteristics:

- The system must be able to use the actual communication systems that exist on the zone.
- If the communications are damaged or does not exist the system must have communication capacities to establish IP reliable communication between the Back-End and the Front-End.
- The communication layer of the system must contemplate redundancy for better adaptation to the different changing scenarios.
- The communication layer of the system must have enough bandwidth and ensure QoS to permit the system operation.

In order to achieve a reliable channel for communication and on the basis of a IP based approach, the most suitable solution could be the usage of tunneling techniques, for example, by means of a Virtual Private Network (VPN) over an available IP backbone. This VPN technique offers a reliable and secure channel over a public network such as Internet. Figure 2.8 explains this concept.

This approach is independent from the technology selected for the communication between the Back-End and the Front-End. The problem here is that how to find a point, to connect an IP backbone which could be physically far away from where the Front-End team is working (as IP backbone close to the disaster scenarios could be down at the moment because of the disaster consequences of the scenario that WORKPAD tries to address). Consequently, the selection of the technology used has to take into account this problem. Especially, in evaluating existing technologies, two critical issues must be covered: coverage and speed transmission offered.

Coverage is already explained above and here the best solutions could be TETRA, GPRS/UMTS or satellite, depending on the availability of such technologies at the moment when the Front-End team arrives at the WORKPAD scenario. Here again the speed transmission issue arises in terms of the capacity of each technology to transmit the amount of data needed to be processed at the Back-End for the Front-End. This is important to define in order to choose the most appropriate solution since technologies such as TETRA offers up to 28.8 kbps while other technologies such as GPRS/UMTS could offer up to 40 kbps to 2 Mbps. Concerning the speed transmission issue another possible solution is to use WiMAX, which does not have the coverage offered by the other but it can offer up to 124 Mbps.

The proposed solution for WORKPAD should be that the Front-End team leader will have several interfaces available for the connection to the Back-End and then choose the most appropriate one by taking into account all the issues described above. The particular nature of the solution of utilizing IP based technologies is that the establishment of a reliable channel by means of a VPN is possible in all cases but the time needed to create such VPN connection has to be considered too.

2.6 Timeline of the WORKPAD Project

The WORKPAD project has had a duration of 36 months; it is effectively started in September 1st, 2006 and ended in August 31st, 2009. 8 different European organizations participated to this project, each one with specific roles inside the project. Specifically:

- *UOR (Italy)* - COORDINATOR AND SCIENTIFIC PARTNER - it was the project coordinator, with a strong research/scientific expertise on Distributed Architectures, Adaptive Process Management on Mobile Adhoc NETworks (MANETs), and P2P Data Integration.

- *TOR VERGATA (Italy)* - SCIENTIFIC PARTNER - has a strong research/-scientific expertise on P2P Architectures and Content Management.

- *IBM ITALIA S.P.A. (Italy)* - INDUSTRIAL PARTNER - has research/scientific expertise on Distributed Architectures, P2P Data Integration and Content Management. It was the main actor in the design and realization of the Back-End.

- *SALZBURG RESEARCH (Austria)* - SCIENTIFIC PARTNER - has research/scientific expertise on geo-information management, usability issues and experience in projects dealing with emergency management and geo-information.

- *TUW (Austria)* - SCIENTIFIC PARTNER - has research/scientific expertise on service oriented systems, context-aware and adaptable systems, distributed architectures.

- *MOVIQUITY (Spain)* - INDUSTRIAL PARTNER - has experience in system integration, in complex communication infrastructures and in software development on mobile devices. MOVIQUITY had a significant role in the design and development of (i) the communication and software infrastructure for setting-up MANETs of devices, and of (ii) the communication infrastructure between Front-End and Back-End.

- *SOFTWARE602 (Czech Republic)* - INDUSTRIAL PARTNER - has expertise in application development on desktops and mobile devices. It will be leading the detailed design and realization of the adaptive process management system for mobile devices.

- *PCRC (Italy)* - USER PARTNER - has expertise in managing emergencies and disasters.

Figure 2.9 helps to understand the work done by each partner during the lifecycle of the Project.

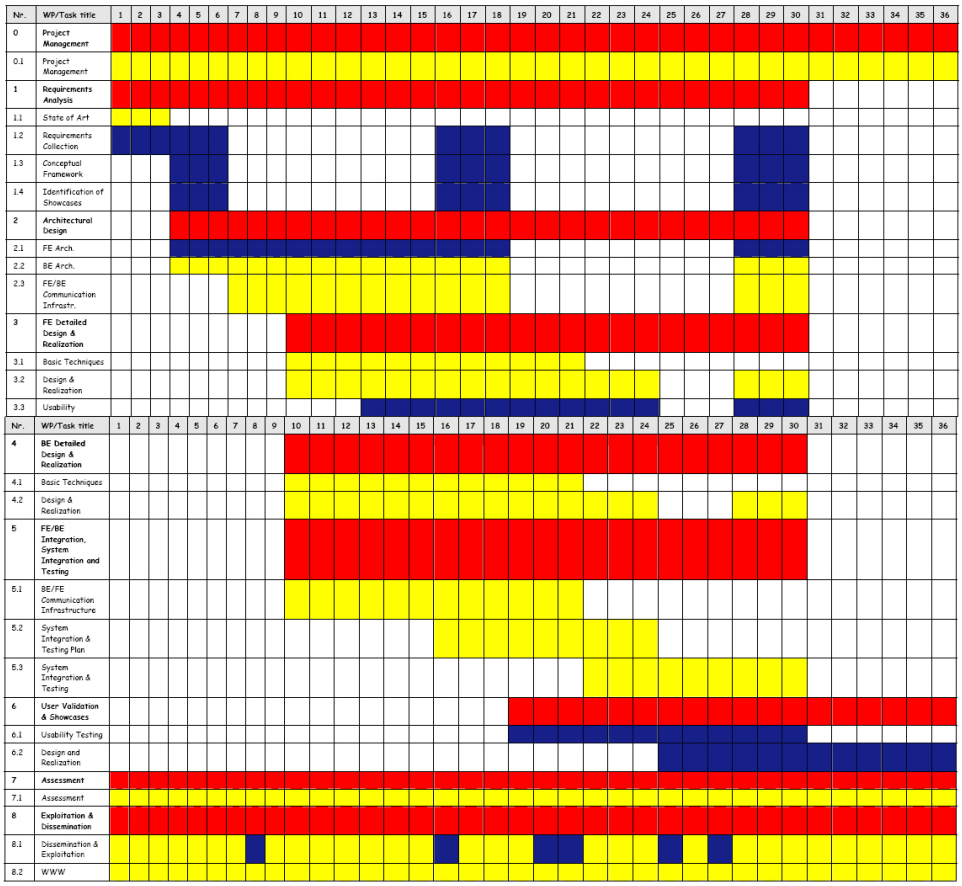


Figure 2.9: The Timeline of the Project - Gantt Diagram

This Section aims to show which stages have been faced by this Thesis during the timeline of the project; in Figure 2.9, the blue rectangles indicates the contributions furnished by this work. Due to the organization of phases of the WORKPAD project, some results have been held after more incremental reviews (for example, the system engineering process was conducted during three iterations). In that cases, for space reasons, this Thesis will report only the final results, obtained in the very last iteration.

Figures 2.10, 2.11, 2.12 depict with major detail the roadmap of the various outcomes held by this work within the project. The violet text boxes in the Figures represent the contributes (results and knowledge) that this Thesis has furnished to the research dissemination.

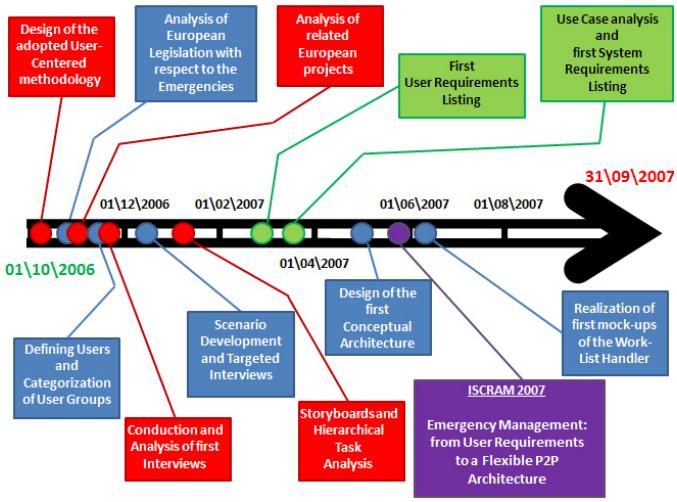


Figure 2.10: Major Outcomes obtained in the First Year

Details about the Roadmap

- **September-October 2006** - *Design of the adopted User-Centered methodology* - Chapter 4.
- **October-November 2006** - *Analysis of European Legislation with respect to the Emergencies* - Section 3.3.
- **October-November 2006** - *Analysis of related European Projects* - Section 3.4.
- **November 2006** - *Defining Users and categorization of Users Groups* - Sections 2.1 and 5.1.
- **November 2006** - *Conduction and Analysis of first Interviews* - Section 5.2.
- **December 2006** - *Scenario Development and targeted Interviews* - Section 5.3.
- **December 2006-January 2007** - *Storyboards and Hierarchical Task Analysis* - Section 5.4.
- **March 2007** - *First User Requirements Listing*.
- **April 2007** - *Use Case Analysis* - Section 5.6.
- **April 2007** - *First System Requirements Listing*.
- **May 2007** - *Design of the first Conceptual Architecture* - Sections 2.2, 2.3, 2.4 and 2.5.

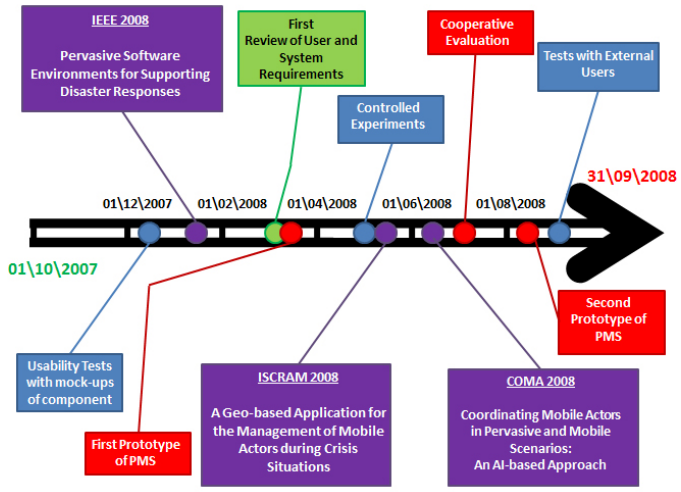


Figure 2.11: Major Outcomes obtained in the Second Year

- **June 2007** - *Realization of first Mock-Up of the WorkList Handler* - Section 7.1.
- **December 2007** - *Performance of a mock-up test with the different WORKPAD components* - Section 6.1.
- **February-April 2008** - *First review of User and System Requirements.*
- **April 2008** - *First working prototype of PMS component.*
- **May 2008** - *Controlled Experiments* - Section 6.2.
- **July 2008** - *Cooperative Evaluation* - Section 6.3.
- **August-September 2008** - *Second prototype of PMS component.*
- **September-October 2008** - *Tests with external Users* - Section 6.4.
- **November 2008** - *Third prototype of PMS component, with adaptivity features.*
- **December 2008** - *SHOWCASE: Test without WORKPAD System* - Section 6.5.
- **February 2009** - *Final prototype of the PMS engine* - Section 7.2.
- **February-April 2009** - *Second review of User and System Requirements* - Sections 5.5 and 5.7.
- **June 2009** - *SHOWCASE: Test with WORKPAD System* - Section 6.6.

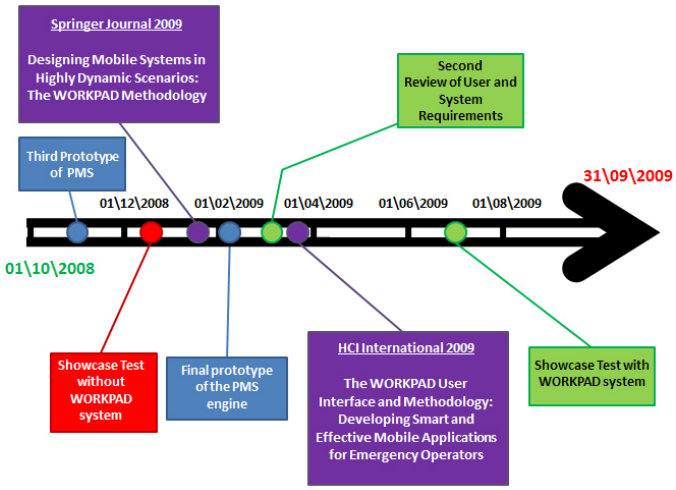


Figure 2.12: Major Outcomes obtained in the Third Year

Chapter 3

State of the Art

The present Chapter provides a State-of-the-Art about the main issues faced in this Thesis. An accurate analysis of the State-of-the-Art represents a good vehicle for the specific purpose to provide powerful and flexible services to mobile cooperating users. However, it is fundamental not only to analyze the highest level of development of the software design techniques, but also to understand how the Emergency Management effectively works in European countries. In fact the WORKPAD project is born with the target to be a system exploitable not only from the main final user (the Italian Civil Protection), but from emergency operators of all European countries too.

Together with Section 2.1, this can be considered as the basis of departure whence all the analysis conducted about WORKPAD project have really started.

The Chapter is organized as follow :

- Section 3.1 lists the plan-driven methodologies used in Software Engineering to impose a disciplined and predictable process upon software development.
- Section 3.2 describes in depth the User-Centered Design techniques.
- Section 3.3 gives an overview about the status of Emergency Management in Europe.
- Section 3.4 analyzes other relevant research projects dealing with emergencies (mostly funded by the European Commission), providing the requirements acquired and the concrete significance for WORKPAD.

3.1 Plan Driven Methodologies

The *plan-driven* is a software development methodology that is used to structure, plan, and control the process of developing an information system. Many frameworks have evolved over the years, each with its own strengths and weaknesses. Each of the available methodologies refer to specific kinds of projects, based on

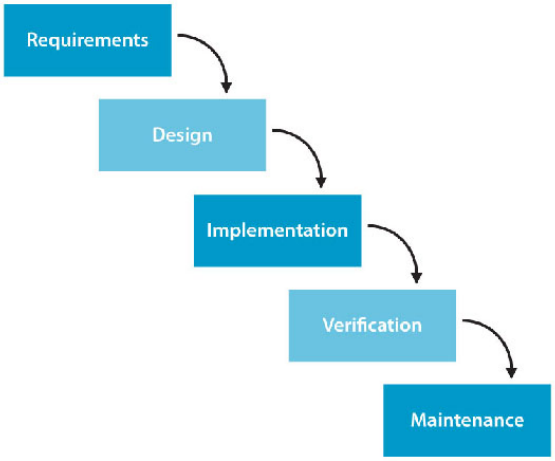


Figure 3.1: The Waterfall model

various technical, organizational, project and team considerations. Every software development methodology has more or less its own approach to software development. There is a set of more general approaches, which are developed into several specific methodologies. These approaches are:

- **Waterfall** (linear)
- **Prototyping** (iterative)
- **Incremental** (combination of linear and iterative)
- **Spiral** (combination of linear and iterative)
- **Rapid Application Development (RAD)** (iterative)

3.1.1 Waterfall model

This is a sequential development process, in which development pass (flowing down like a waterfall) through the phases of requirements analysis, design, implementation, testing (validation),integration, and maintenance (see Figure 3.1)

The first formal description of the Waterfall Model is often cited to be an article published in 1970 by Winston W. Royce [45], although Royce did not use the term "waterfall" in this article and described the simplest form as "risky and invites failure".

Basic principles of the Waterfall Model are:

- Emphasis is on planning, time schedules, budgets, target dates and implementation of an entire system at one time.

- Tight control is maintained over the life of the project through the use of extensive written documentation. Moreover formal reviews and approval by the users and information technology management occur at the end of most phases.

Thanks to its linear structure, this methodology is characterized by some points of strengths, such as the reduction of planning overhead (since it can be done up front) and the minimization of wasted effort (it works well for technically weak or inexperienced staff). But, in general, the Waterfall Model is argued by many to be a bad idea in practice, mainly because of their belief that it is impossible, for any non-trivial project, to get one phase of a software product's life-cycle perfected before moving on to the next phases and learning from them. For example, clients may not be aware of exactly what requirements they want before they see a working prototype and can comment upon it; they may change their requirements constantly, and program designers and implementers may have little control over this. If clients change their requirements after a design is finished, that design must be modified to accommodate the new requirements, invalidating quite a good deal of effort invested in the phase before.

3.1.2 Prototyping

Software Prototyping is a set of activities held during software development for creating prototypes (i.e., incomplete versions of the software program being developed). A prototype typically simulates only a few aspects of the features of the eventual program, and may be completely different from the final implementation. Introduced in 1975 by Fred Brooks [10], the conventional purpose of a prototype is to allow users of the software to evaluate developers' proposals for the design of the eventual product by actually trying them out, rather than having to interpret and evaluate the design based on descriptions. Prototyping can also be used by end users to describe and prove requirements that developers have not considered, so "controlling the prototype" can be a key factor in the commercial relationship between solution providers and their clients.

Basic principles of Prototyping are:

- Not a standalone methodology, but rather an approach to deal whit portions of a more traditional development methodology.
- Attempts to reduce project risk by breaking a project into smaller segments.
- User is involved throughout the process, which increases the likelihood of user acceptance of the final implementation.
- Small-scale mock-ups of the system are developed following an iterative modification process until the prototype evolves to meet the user's requirements.

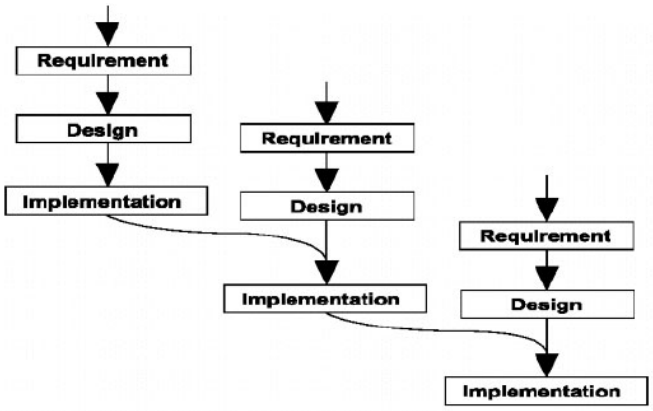


Figure 3.2: The Incremental model

- Most prototypes are developed with the expectation that they will be discarded, but it is possible in some cases to evolve from prototype to working system.
- It is impossible to know at the outset of the project how long it will take.
- There is no way to know the number of iterations that will be required.
- The start up costs for building a development team focused on prototyping may be high.

It has been argued that Prototyping should be used all the time; however, Prototyping is most beneficial in systems that will have many interactions with the users. The greater the interaction between the computer and the user, the greater the benefit is that can be obtained from building a quick system and letting the user play with it [18]. In particular, Prototyping is especially good for designing good Human-Computer Interfaces [3].

3.1.3 Incremental model

Incremental development is a scheduling and staging strategy, in which the various parts of the system are developed at different times or rates and integrated when they are completed. It does not imply, require or not preclude iterative or Waterfall development (see Figure 3.2). The product is decomposed into a number of components, each of which are designed and built separately (termed as builds). Each component is delivered to the client when it is complete. This allows partial utilization of product and avoids a long development time. It also creates a large initial capital outlay with the subsequent long wait avoided. This model of development also helps ease the traumatic effect of introducing completely new system all at once.

Basic principles of Incremental development are:

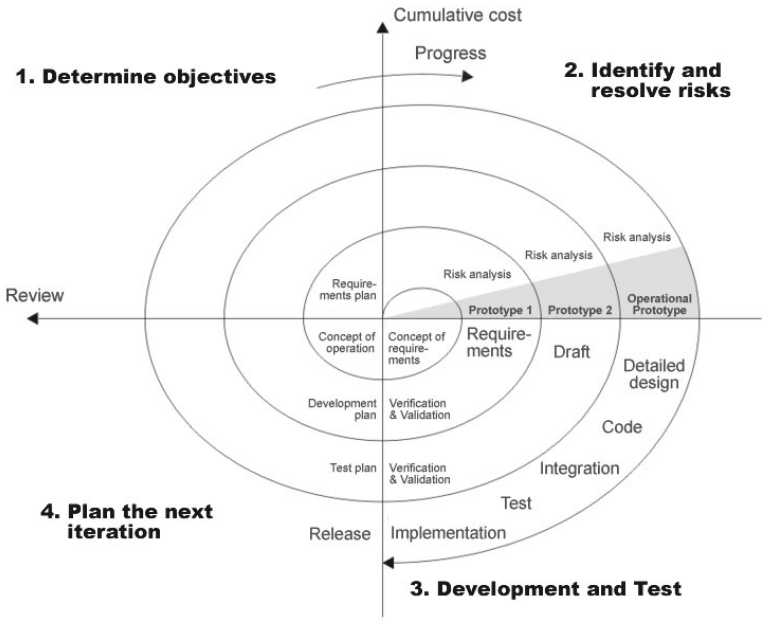


Figure 3.3: The Spiral model

- A series of mini-Waterfalls are performed, where all phases of the Waterfall Model are completed for a small part of the systems, before proceeding to the next incremental.
- Requirements are defined before proceeding to evolutionary, mini-Waterfall development of individual increments of the system.
- The initial software concept, requirements analysis, and design of architecture and system core are defined using the Waterfall approach, followed by iterative Prototyping, which culminates in the installation of the final prototype.

There are some problems with this model. One is that each new build must be integrated with previous builds and any existing systems. If there are few builds and each build degenerates, this turns into Build-And-Fix model. However, if there are too many builds, then there is little added utility from each build.

3.1.4 Spiral model

The Spiral Model is a software development process combining elements of both design and Prototyping in stages, in order to combine advantages of top-down and bottom-up concepts; it combines the features of the Prototyping Model and the Waterfall Model. The Spiral Model was defined by Barry Boehm in 1988 [9] and it is intended for large, expensive and complicated projects.

Basic principle of Spiral development are:

- Focus is on risk assessment and on minimizing project risk by breaking a project into smaller segments and providing more ease-of-change during the development process. It gives the opportunity to evaluate risks of project throughout the life cycle.
- Each cycle requires a progression through the same sequence of steps for each portion of the product; it begins from an overall concept-of-operation document down to the coding of each individual program.
- It promotes quality assurance through prototyping at each stage in systems development.
- The software is produced early in the life cycle.
- Begin each cycle with an identification of stakeholders and their requests, and terminate each cycle with review and commitment.
- Each trip around the spiral traverses four basic quadrants: (1) determine objectives, alternatives, and constraints of the iteration; (2) identify and resolve risks; (3) develop and verify deliverables from the iteration; and (4) plan the next iteration.

The steps in the Spiral Model can be generalized as follows (see Figure 3.3):

1. The new system requirements are defined in as much detail as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
2. A preliminary design is created for the new system. This phase is the most important part of Spiral Model. In this phase all possible (and available) alternatives, which can help in developing a cost effective project are analyzed and strategies are decided to use them. This phase has been added specially in order to identify and resolve all the possible risks in the project development. If risks indicate any kind of uncertainty in requirements, Prototyping may be used to proceed with the available data and find out possible solution in order to deal with the potential changes in the requirements.
3. A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
4. A second prototype is evolved by a fourfold procedure:
   (a) evaluating the first prototype in terms of its strengths, weaknesses, and risks;
   (b) defining the requirements of the second prototype;
   (c) planning and designing the second prototype;
   (d) constructing and testing the second prototype.

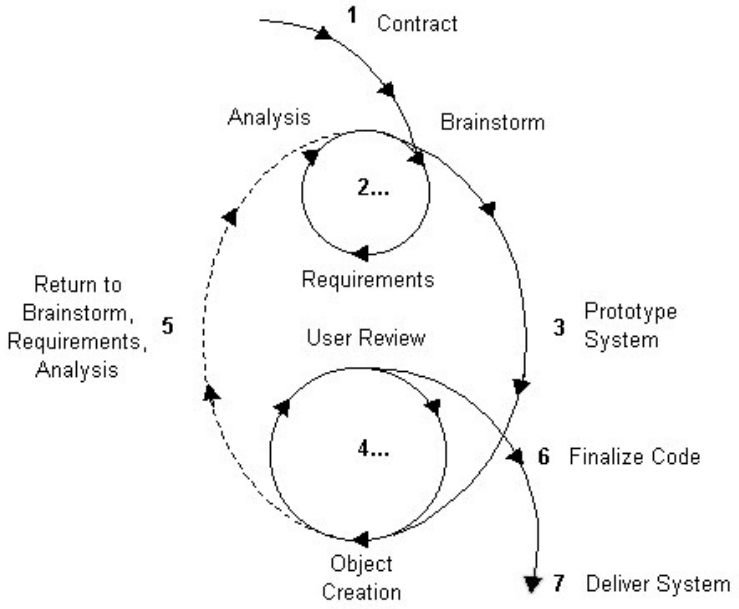


Figure 3.4: The Rapid Application Development (RAD) model

3.1.5 Rapid Application Development (RAD)

Rapid Application Development is a term originally used to describe a software development process introduced by James Martin in 1991 [40]. Martin's methodology involves iterative development and the construction of prototypes (see Figure 3.4). More recently, the term and its acronym have come to be used in a broader, generic sense that encompasses a variety of techniques aimed at speeding application development, such as the use of web application frameworks and other types of software frameworks. Therefore RAD refers to a type of software development life cycle which uses minimal planning in order to a rapid prototyping. The planning of software written in RAD is interleaved with writing the software itself. The lack of extensive pre-planning generally allowed software to be written much faster, and makes it more flexible to change in requirement.

Basic principles of Rapid Application Development are:

- Key objective is for fast development and delivery of a high quality system at a relatively low investment cost.
- Reduce project risk by breaking a project into smaller segments and providing more ease-of change during the development process.
- Produce high quality systems quickly, primarily through the use of iterative Prototyping (at any stage of development), active user involvement, and computerized development tools.

- Active user involvement is imperative.
- Produces documentation necessary to facilitate future development and maintenance.

3.2 User-Centered Design

The design of everyday objects is not always intuitive and it often leaves the user frustrated and unable to complete a simple task. User-Centered Design (UCD) is a broad term to describe design processes in which end-users influence how a design takes shape. So UCD is both a broad philosophy and a variety of methods in which the needs, wants, and limitations of the end user are taking into account at each stage of the design process. The main difference from other interface design philosophies is that UCD tries to optimize the user interface around how people can, want, or need to work, rather than forcing the users to change how they work to accommodate the software developers approach. There is a spectrum of ways in which users are involved in UCD but the important concept is that users are involved one way or another. For example, some types of UCD consult users about their needs and involve them at specific times during the design process; typically during requirements gathering and usability testing. At the opposite end of the spectrum there are UCD methods in which users have a deep impact on the design by being involved as partners with designers throughout the design process.

The term "User-Centered Design" was originated in Donald Norman's research laboratory at the University of California San Diego in the 1980 and became widely used after the publication of the book [19]. The role of the designer is to facilitate the task for the user and to make sure that the user is able to make use of the product as intended and with a minimum effort to learn how to use it. For example, Norman noted that the long cumbersome, unintelligible manuals that accompany products are not User-Centered. He suggests that the products should be accompanied by a small pamphlet that can be read very quickly and draws on the user's knowledge of the world. Telling designers that products should be intuitive is not enough; some design principles are needed to guide the design. Norman suggested that the following seven principles of design are essential for facilitating the designer's task:

1. Use both knowledge in the world and knowledge in the head. By building conceptual models, write manuals that are easily understood and that are written before the design is implemented.

2. Simplify the structure of tasks. Make sure not to overload the short-term memory, or the long term memory of the user. On average the user is able to remember five things at a time. Make sure the task in consistent and provide mental aids for easy retrieval of information from long-term memory. Make sure the user has control over the task.

3. Make things visible: bridge the gulfs of Execution and Evaluation. The user should be able to figure out the use of an object by seeing the right buttons or devices for executing an operation.

4. Get the mappings right. One way to make things understandable is to use graphics.

5. Exploit the power of constraints, both natural and artificial, in order to give the user the feel that there is one thing to do.

6. Design for error. Plan for any possible error that can be made, this way the user will be allowed the option of recovery from any possible error made.

7. When all else fails, standardize. Create an international standard if something cannot be designed without arbitrary mappings.

In 1987 Ben Shneiderman articulated a similar set of principles in the form of eight golden rules [47]. Later Jakob Nielsen adapted and popularized these same basic concepts to produce heuristics for usability engineering [42]. Norman's work stressed the need to fully explore the needs and desires of the users and the intended uses of the product. The need to involve actual users, often in the environment in which they would use the product being designed, was a natural evolution in the field of User-Centered design. Users became a central part of the development process. Their involvement lead to more effective, efficient and safer products and contributed to the acceptance and success of products [31].

3.2.1 The phases of UCD

User-Centered Design can be characterized as a multi-stage problem solving processes that not only requires designers to analyze and foresee how users are likely to use an interface, but also to test the validity of their assumptions with regards to user behavior in real world.

The international standard ISO 13407 [1] is a general reference model of User-Centered Design principles and processes, and describes them as follows:

- **Focus on end-users.** The key principle of User-Centered design is to place the users at the centre of the design activities: the users' tasks, user participation and user evaluation summarize the essence of the approach. The early and active involvement of users helps designers to avoid unpromising design paths and develop a deeper understanding of the actual problem scope. Users can either be actively or passively involved in the system design process. Passive user involvement is the case where users simply state their needs and requirements. Actively involved users participate in the design process by being actively integrated into the system design. Any user involvement will increase the likelihood of user satisfaction with the final application. End-users can contribute to the system design by providing information about:

1. Their knowledge of the task domain;
2. Their prior experiences with applications which they have used or are currently using;
3. Their work objectives and responsibilities, tasks and workflows;
4. Their subjective requirements for applications.

Users are not only consulted at the beginning of the development process. They are invited to evaluate the system at the end and are treated as partners throughout the whole design and development process.

- **Iterative and incremental design.** with a conjoint use of approaches describes in Sections 3.1.2, 3.1.3 and 3.1.4.

- **Appropriate allocation of functionalities between end-users and the (computer) system.** An appropriate allocation of functionalities aims at ensuring that the human can carry out meaningful tasks on the human side of the human-computer partnership. How the human conducts his part of the tasks has to be considered in the overall design of the system, which is referred to as "user modeling". Within the system design process, it must be specified which functionalities shall be carried out by the system and which by the potential users. This task is performed taking into account limited human information processing capabilities and limited performance of the technology in terms of reliability, speed, accuracy, flexibility of response, cost, importance of successful or timely accomplishment of tasks etc.

Moreover, the ISO standard identifies four iterative design activities. Figure 3.5 depicts the basic UCD activities.

Phase 1: Understanding of Users

The UCD process relies on iterative user research to understand users, their needs and their context of use. The successful design of a product must take into account the wide range of users of the artefact and how to involve them in the design process. Not everyone who is a "stakeholder" needs to be represented on a design team, but the effect of the artefact on them must be considered [31]. For example, in [23] were identified three types of users:

1. Primary users are those persons who actually use the artefact.

2. Secondary users are those who will occasionally use the artefact or those who use it through an intermediary.

3. Tertiary users are persons who will be affected by the use of the artefact or make decisions about its purchase.

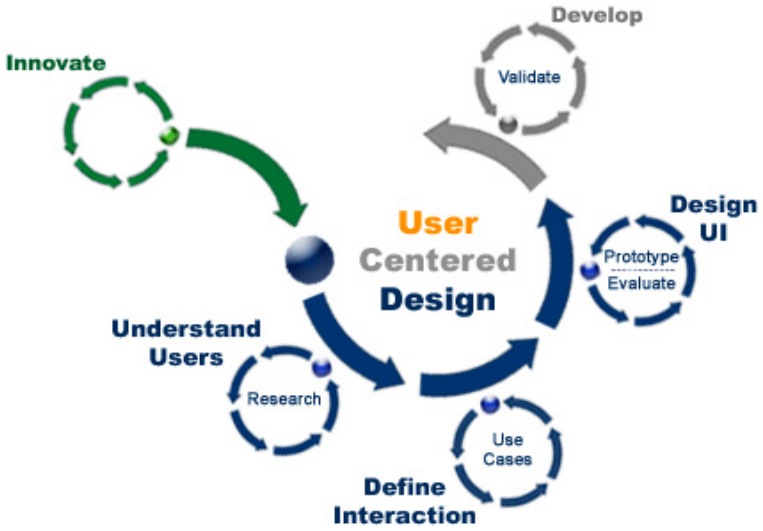


Figure 3.5: The phases of UCD

Once the stakeholders have been identified and a thorough investigation of their needs has been conducted by performing tasks and needs analysis, designers can develop alternative design solutions to be evaluated by the users. These design solutions can be simple paper and pencil drawings in the beginning phase of the process. Listening to users discuss the alternative designs can amplify designers understanding of the intended purpose(s) of the artefact and may provide information that does not come out of initial interviews, observations, and needs analysis.

Phase 2: Define Interaction

The most common failure point in UCD processes is transferring the understanding of users to User Interface design. The key is to define interaction first, without designing it. First, all the user research conducted is organized and summarized in a user research synthesis, leading to user profiles, work activities, and requirements for the intended user populations. The summarized user research information feeds directly into use cases, which define a products use. To start use cases, a subset of work activities are identified and organized into a coherent product with a high-level overview on how information will flow throughout the application. Then the specified work activities are captured in further detail with goal-based use cases. The use cases show steps to accomplish task, goals and the data needed to perform interactions. The data definitions are the only elements of an interface that need to be determined in this phase. Completed use cases are validated with the intended user population. This is a checkpoint to see if the vision is being achieved and the value is clear to the users.

Phase 3: Design User Interface (UI)

The third phase of UCD is to design the User Interface (UI), evolving directly from the interaction definition.. Product scope and interface organization are clear from the high-level information organization, and UI components are clear from use case steps and data. A primary concern with design is to not get locked into a single solution too early. To help prevent design traps, this phase is explicitly broken into two stages; low-fidelity Prototyping and high-fidelity Prototyping. Low-fidelity prototypes allow experimentation and rapid evaluation. High-fidelity prototypes provide exacting design and behavior previews of the final product that specifies what is to be coded.

Phase 4: Development Validation

At this point, designers should pay close attention to the evaluations by the users as they will help identify measurable usability criteria. Measurable usability criteria address issues related to the effectiveness, efficiency, safety, utility, learnability and memorability (how long it takes to remember to perform the most common tasks) of the product/artefact and user's subjective satisfaction with it. It is only through feedback collected in an interactive iterative process involving users that products can be refined. Lastly, completed products are formally tested for usability in benchmark usability test. These tests monitor ongoing improvement over time, against prior products and versions, as well as the competition.

3.2.2 How to Involve Users in Design?

It is necessary to think carefully about who is a user and how to involve users in the design process. There are some methods can be used to detect the end-users of an application and to understand which are their real needs and requests. This section suggests some techniques to involve users in the design and development of a product/artifact, analyzing their pros and cons and the number of users requested to be applied.

Interviews

Interviewing [39] is a HCI technique for getting a clearer picture of the users and their tasks and goals. For this, it is highly beneficial to "define" the potential user of the system and ideally to introduce user groups - especially if there is a large number of heterogeneous users. These groups can then be treated separately if necessary and different conclusions may be drawn. Interviewing involves structured or unstructured discussions between requirement engineers and potential users of a future system. Interviews can be conducted most effectively if the requirements engineer already has a basic understanding about the user context. This understanding can be gathered through some background research or simply via asking relevant people a priori. The interviews should then be designed to get clear and

focussed answers that can subsequently serve as a basis for the further design and development phases. These are usually referred to as structured interviews. In contrast, unstructured interviews can be used if the requirements engineer wants to have an open-end discussion with the users. Unstructured interviews, moreover, can also serve for the first getting in touch with the problem context. Interviews can also be used to communicate common and variable requirements that are already gathered by the users so that further negotiations can take place [39]. Thereafter, the results of the interviews are analyzed which hopefully leads to new insights. Usually, the analysis of structured interviews with closed questions is easier as the answers are clearly defined. Unstructured interview analysis, however, is probably more time-consuming but depending on the quality of the interview conduction and the answers.

* **Number of Users :** Varies (few-many).
* **Stage of the Design Cycle :** Useful in each stage of design project.
* **Pros and Cons :**

  (+) low-cost, direct way to gather data;

  (+) effective for identifying users'needs and opinions;

  (-) will not reveal any information that subjects wish to remain hidden or that they don't consciously know;

  (-) dependent on participant's memory and willingness to contribute;

  (-) can be challenging to schedule with busy users;

  (-) can be challenging to both record data and facilitate the interview (audio-taped transcripts are recommended for long interviews).

Survey

A survey is a instrument consisting of a series of questions and other prompts for the purpose of gathering information from respondents. A distinction is made between open-ended and closed-ended questions. An open-ended question asks the respondent to formulate his own answer, whereas a closed-ended question has the respondent pick an answer from a given number of options. The response options for a closed-ended question should be exhaustive and mutually exclusive. Four types of response scales for closed-ended questions are distinguished:

- Dichotomous, where the respondent has two options;
- Nominal-polytomous, where the respondent has more than two unordered options;
- Ordinal-polytomous, where the respondent has more than two ordered options(bounded);

- Continuous, where the respondent is presented with a continuous scale.

A respondent's answer to an open-ended question is coded into a response scale afterwards.

* **Number of Users :** Varies (few-many).

* **Stage of the Design Cycle :** Useful in each stage of design project.

* **Pros and Cons :**

  (+) low-cost;

  (+) can gather data from many users quickly;

  (+) can have standardized answers that make it simple to compile data;

  (-) dependent on users choosing to participate in the survey;

  (-) standardized answers may frustrate users;

  (-) can be sharply limited by the fact that respondents must be able to read the questions and respond to them;

  (-) most valuable questions (open-ended) are often left unanswered.

Focus Groups

Focus groups are another HCI technique used to learn to understand the user [35]. A focus group meeting is very concentrated and comprises only a maximum of eight people. Basically, it can be distinguished between three phases :

- preparation,

- conduction, and

- post-processing.

The better the meeting is prepared the better the discussions might be and the better are the results. The moderator has to have a clear picture of what shall be achieved and has to have a clear goal which also must be stated at the beginning of the actual meeting. After some inputs of the moderator, several issues shall be discussed whereby the added-value arises through group dynamics. It is essential that one person is the dedicated recorder, such that no results are lost. The moderator should be rather passive in discussions and should try to organize and coordinate the inputs of the participants. In the post-processing phase the discussions are summarized and the main results derived. These shall again be circulated among the participants in order to get some further details or to avoid misunderstandings.

* **Number of Users :** 6-8 users.

* **Stage of the Design Cycle :** Early in the design cycle.

* **Pros and Cons :**

  (+) raises objections and insecurities regarding a system or its use that might not be discovered through other means;

  (+) can generate large amounts of data in a relatively short time;

  (-) requires an experienced facilitator;

  (-) subject to "domination effect" in which one participant sways the discussion to a single point of view;

  (-) subject to known inconsistencies between what people will say in a group and what their actual behavior may be.

Scenarios

The term "scenario" is somehow overloaded and may mean several things. The notion of scenario in the fields of HCI, object-oriented software engineering, and requirements engineering ranges from real world descriptions and stories in a rather informal and narrative style (also with the use of pictures) to formal models and specifications (structured models, tables, UML diagrams etc). Also the scope may range from a very general, abstract, broad, and high-level description to very concrete facets of real world situations such as a single example of an event sequence. Scenarios can be exploited for a great range of different usages, such as requirements analysis, user-designer communication, for examples to motivate design rationale, envisionment (to help imagine a future system), software design, implementation, training, and documentation [52].

* **Number of Users :** Varies (few-many).

* **Stage of the Design Cycle :** Useful in each stage of design project.

* **Pros and Cons :**

  (+) simulates a possible real situation;

  (+) allows users to dip completely themselves into the situation to analyze;

  (-) can be dispersive;

  (-) is really time consuming.

Storyboards

Prior to conducting task analysis, concrete data needs to be available. One way to provide such data are storyboards. Storyboards are a kind of interpretation of a scenario where a concrete walk through of several events can be conducted and key steps in the process are highlighted [52]. Due to the higher level of detail, one storyboard can never cover a whole scenario. Thus, several storyboards should be developed to subsume the most important aspects of a scenario. These storyboards,

in turn, can be used as the major input to conduct task analysis. It is the study of the way people perform tasks broken down into further subtasks. This technique aims at showing an overall structure of the main user tasks, underlining three features:

1. What users do;
2. Which things they work with;
3. What they must know to perform the task.

This information can then be used for many purposes, such as improving the design of tools or procedures that aid in performing the task.

* **Number of Users :** Varies (few-many).
* **Stage of the Design Cycle :** Useful in each stage of design project.
* **Pros and Cons :**

  (+) focuses on specific situations of a Scenario;

  (-) is really time consuming.

Task Analysis

Task analysis does not focus on the cognitive processes of the user while a task is performed, but only on the strategy carried out to reach the goal, seen as a sequence of steps. Therefore, visible behaviors of the users are observed and it is analyzing what users need to do when performing the task - and not why. It does not describe the optimal or ideal procedure for solving a problem, but it simply describes the way in which the problem is currently solved. Task analysis is often confounded with system analysis, which is a description of how a system reacts to a task performed by the user interacting with it. The main difference to this technique is that in task analysis the user is placed at the center of the model, while in system analysis it is focused on the behavior of the system. By using task analysis the difference a system engineer's assumptions and the users' mental models of a task may become apparent more easily. Task Analysis is mostly done in the form of a hierarchical description of the main tasks which are then separated into several levels of subtasks [3]. This approach has been introduced by [6] and referred to as Hierarchical Task Analysis (HTA). Special emphasis should be paid to the tasks for which explicit requirements are set in the user requirements process.

* **Number of Users :** At least 5 users/groups.
* **Stage of the Design Cycle :** Early in the design cycle.
* **Pros and Cons :**

(+) can reveal new information that is exploitable in the software design (e.g. short cuts that expert users take);

(+) fundamental to understand which task is "atomic" in the mind of user;

(-) can be time consuming to carry out;

(-) if not observing an expert user, you can inadvertently reproduce an inefficient way to complete a task;

(-) if you do observe an expert user, you may not find out the problems specific to beginners.

The following three methods can be used for the task analysis: task decomposition, knowledge-based methods, or entity-relation based techniques. They are described below:

- **Task Decomposition** The aim of high-level task decomposition is to decompose high-level tasks and break them down into their constituent subtasks and operations. A task can be defined as a goal combined with some ordered set of actions. The process of task decomposition is best represented as a structure chart, which demonstrates the sequencing of activities by ordering them from left to right. In order to break down a task, the question should be asked "how is this task done?". If a sub-task is identified at a lower level, it is possible to build up the structure by asking "why is this done?". The task decomposition can be carried out using the following stages as suggested by [33]:

  1. Identify the task to be analyzed.
  2. Break this task down into a certain number of subtasks. These subtasks should be specified in terms of objectives and should cover the whole area of interest.
  3. Draw the subtasks as a layered diagram ensuring that it is complete.
  4. Decide upon the level of detail into which to decompose. Making a conscious decision at this stage will ensure that all the subtask decompositions are treated consistently. It may be decided that the decomposition should continue until flows are more easily represented as a task flow diagram.
  5. Continue the decomposition process, ensuring that the decompositions and numbering are consistent. It is usually helpful to produce a written account as well as the decomposition diagram.
  6. Present the analysis to someone else who has not been involved in the decomposition but who knows the tasks well enough to check for consistency.

  This phase helps software developers to provide a helpful toolkit for understanding everyday processes and for describing how human beings solve

problems. The results of a task analysis can beneficially contribute to the design or pinpoint usability problems.

- **Knowledge-based Techniques** Knowledge-based techniques describe objects and actions that are relevant for the task and are based on the analysis of the user's knowledge about tasks: What does the user know about tasks and about the way they are organized? The knowledge that is needed for this technique can be gathered from the following sources:

  – Documentation: how people are supposed to do it, how does the system work;

  – Observation: procedural analysis vs. knowledge-based analysis;

  – Interviews;

  – Initial analysis and studies;

  – Annotated list of objects and actions;

  – Sorting and/or classification.

  Knowledge-based techniques build on ontology (hierarchical knowledge description) of the selected elements. The objects are then arranged into groups based on similarity or shared traits. The traits that are used to determine groupings depend entirely upon the task that is being analyzed and the purpose of the analysis.

- **Entity-relationship-based Analysis** Entity-relationship-based analysis deals with objects, actions, and their relationships. It divides tasks into simple objects, actors, composite objects, and events. Each simple object can have attributes or qualities associated with it, for example size, colour, or weight. Actors have attributes, but are also associated with them. Actors perform actions upon objects, possibly changing the attributes of the object in the process. Events are things that take place spontaneously, either randomly or caused by actions outside the model, which objects and actors react to. This analysis is a bottom-up approach to task analysis and inherits much of its structure and analysis methods from object-oriented programming.

Usability Testing

Usability testing usually involves systematic observation under controlled conditions to determine how well people can use the product. The aim is to observe people using the product to discover errors and areas of improvement. Usability testing generally involves measuring how well test subjects respond in four areas: efficiency, accuracy, recall, and emotional response. The results of the first test can be treated as a baseline or control measurement; all subsequent tests can then be compared to the baseline to indicate improvement.

- **Performance** How much time, and how many steps, are required for people to complete basic tasks? (For example, find something to buy, create a new account, and order the item).
- **Accuracy** How many mistakes did people make? (And were they fatal or recoverable with the right information?)
- **Recall** How much does the person remember afterwards or after periods of non-use?
- **Emotional** response How does the person feel about the tasks completed? Is the person confident, stressed? Would the user recommend this system to a friend?

Setting up a usability test involves carefully creating a scenario, or realistic situation, wherein the person performs a list of tasks using the product being tested while observers watch and take notes. Several other test instruments such as scripted instructions, paper prototypes, and pre- and post-test questionnaires are also used to gather feedback on the product being tested.

* **Number of Users :** At least 5 users.
* **Stage of the Design Cycle :** Final stage of the design cycle.
* **Pros and Cons :**
  - (+) a small number of users can identify numerous problems in a relatively short amount of time;
  - (+) finds more authentic problems than inspection methods;
  - (-) user's performance may be affected depending on the perceived unreality of the session, their nervousness, and the effect of being observed;
  - (-) the meaningfulness of the data collected rests on the authenticity of the users and tasks involved;
  - (-) time consuming to plan and analyze.

Expert Review

Design experts examine the system or a prototype of it and comment in detail on its adherence to principles of good design based on their expertise. Multiple experts are recommended to increase the probability that they will identify the main problems.

* **Number of Experts :** 1-4 experts.
* **Stage of the Design Cycle :** Useful starting from mid-point of the design cycle.

* **Pros and Cons :**

  (+) resolves some issues that users should not have to worry about in later usability testing;

  (-) constrained by the expert's knowledge of the audience for which the system is intended;

  (-) not sufficient on its own (developers will not catch the same problems as users will).

Guided Walkthrough

The method is rooted in the notion that users typically prefer to learn a system by using it to accomplish tasks, rather than, for example, studying a manual. The method is prized for its ability to generate results quickly with low cost, especially when compared to usability testing, as well as the ability to apply the method early in the design phases, before coding has even begun. Guided Walkthrough starts with a task analysis that specifies the sequence of steps or actions required by a user to accomplish a task, and the system responses to those actions. The designers and developers of the software then walkthrough the steps as a group, asking themselves a set of questions at each step. Data is gathered during the walkthrough, and afterwards a report of potential issues is compiled. Finally the software is redesigned to address the issues identified.

* **Number of Users :** Varies (few-many).

* **Stage of the Design Cycle :** Useful in each stage of the design cycle.

* **Pros and Cons :**

  (+) does not require a high-fidelity prototype;

  (+) can reveal attitudes and expectations that the user might not otherwise express;

  (-) must be conducted carefully to avoid accidentally leading users to conclusions or misinterpreting their actions;

  (-) subjective point of view will keep developers from recognizing some problems;

  (-) user performance may be effected by observation, nervousness, or other assessment factors.

Heuristic Evaluation

It specifically involves evaluators examining the interface and judging its compliance with recognized usability principles (the "heuristics"). Multiple experts are recommended to increase the probability that they will identify the main problems.

* **Number of evaluators :** 1-5 users.

* **Stage of the Design Cycle :** Mid-Point and final stage of the design cycle.

* **Pros and Cons :**

  (+) inexpensive, quick, and easy way to identify usability problems;

  (-) possible to identify usability problems that actually may not be a problem of the user;

  (-) constrained by the evaluator's knowledge of HCI and knowledge of the audience for which the system is designed;

  (-) not sufficient on its own (developers will not catch the same problems as users will).

3.2.3 Conclusions

The major advantage of the User-Centered Design approach is that a deeper understanding of the psychological, organizational, social and ergonomic factors that affect the use of computer technology emerges from the involvement of the users at every stage of the design and evaluation of the product. The involvement of users assures that the product will be suitable for its intended purpose in the environment in which it will be used. This approach leads to the development of products that are more effective, efficient, and safe.

It also helps designers manage user's expectations about a new product. When users have been involved in the design of a product, they know from an early stage what to expect from a product and they feel that their ideas and suggestions have been taken into account during the process. This leads to a sense of ownership for the final product that often results in higher customer satisfaction and smoother integration of the product into the environment [31].

If the design is not user-centered, it could lead to uncorrected designs. When user's expectations are not met, they may get frustrated. The major disadvantage to UCD is that is can be quite costly. It takes time to gather data from and about users especially if you seek to understand the environment in which they will be using the products. The process requires resources, both financial and human. User-Centered Design teams generally benefit from including persons from different disciplines, particularly psychologists, sociologists and anthropologists whose job it is to understand users needs and communicate them to the technical developers in the team. The downside of this approach is that members of the team have to learn to communicate effectively and to respect each other's contributions and expertise. This can be time consuming and hence adds costs to the process. Management may question whether this added value is worth the cost, particularly if delivery dates are threatened [31, 3]. Figure 3.6 summarizes these and other advantages and disadvantages of UCD.

Advantages	Disadvantages
Products are more efficient, effective, and safe	It is more costly
Assists in managing users' expectations and levels of satisfaction with the product	It takes more time
Users develop a sense of ownership for the product	May require the involvement of additional design team members (i. e. ethnographers, usability experts) and wide range of stakeholders
Products require less redesign and integrate into the environment more quickly	May be difficult to translate some types of data into design
The collaborative process generated more creative design solutions to problems.	The product may be too specific for more general use, thus not readily transferable to other clients; thus more costly

Figure 3.6: Advantages and disadvantages of User-Centered Design

3.3 European Legislation with respect to Emergencies

The harmonization of national legislation dealing with major disasters is regarded as a strategic objective in the European Union. Disasters are not restricted to national borders and often have trans-boundary consequences. The European Commission has the responsibility to support and to supplement efforts at national, regional and local level with regard to

- emergency prevention;
- the preparedness of those responsible for emergency management;
- the intervention in the event of emergency.

The legislative framework for European emergency management allowed the Commission to establish a framework for effective and rapid cooperation between national emergency management services when mutual assistance is needed.

This Section presents the Civil Protection legislation in the European Union, describes emergency management in different European countries and draws a conclusion concerning WORKPAD requirements.

3.3.1 State of the Art in Civil Protection Legislation

The European Union initiated actions in the field of Civil Protection coordination which the goal to support and supplement efforts at national, regional and local levels with regard to disaster prevention. This subsection gives an overview of the legislative framework that governs European emergency management assistance. It is largely based on three pieces of European legislation:

1. the Council Decision of 23 October 2001 established the *Community Civil Protection mechanism for emergency management*. It has the goal of establishing a framework for effective and rapid cooperation between national

Civil Protection services; on the one hand when assistance is needed and on the other hand to enhance the coherence of actions undertaken at international level in the field of civil protection. The treaty has an intergovernmental character and the responsibility of the Community is limited to support and coordinate actions or to amend the work of the member states. The Community Civil Protection Mechanism involves the participation of 30 European states which pool their civil protection resources that can be made available to disaster-stricken countries. The Community Civil Protection mechanism can be activated in case of natural and man-made disasters, including nuclear incidents [17]. The community mechanism itself provides certain tools for the adoption in emergency management [54]. These are:

- Monitoring and Information Center (MIC).
- Common Emergency and Information System (CECIS).
- Training Program.

2. *Proposal for Council Regulation* (20th April 2004) establishes a Rapid Response and Preparedness Instrument for major emergencies. This document serves as the future legal framework for the financing of emergency management operations.

3. *Community Action Programme* (1999-2006): in the field of emergency management this programme intends to support and supplement member states efforts at national, regional and local levels by implementing actions for the protection of persons, property and environment in the event of natural and technological disasters. The programme includes initiatives dealing with prevention, preparedness and response to disasters, as well as information and awareness raising activities.

It is interesting to give a further glance to the Community Civil Protection Mechanism Tools adopted by European Union in emergency management.

- ***Monitoring and Information Center (MIC)*** The Monitoring and Information Centre (MIC) is the operational heart of the community mechanism. The MIC is managed by the Civil Protection Unit of the Directorate General for Environment at the European Commission and has been operational since January 2002. So far the MIC has dealt with more than 50 emergencies, and specialized personnel who are on stand-by 24 hours a day seven days a week is employed into it. The MIC provides countries access to a platform of emergency management means available amongst all the participating countries; any country inside or outside the Union which is affected by a major emergency can make an appeal for assistance through the MIC.

  The MIC also plays a coordination role in case of emergency; when a country hit by a disaster asks for help, the MIC forwards the request for assistance to the contact points of all the other participating countries. Then they process

the responses and send them back to the country which needs assistance. The participating states offer bilateral civil protection assistance to the requesting state. Additionally, a training program has been installed to improve the preparedness of the experts and team leaders. In this way, within a few hours, the requesting country has an overview of all the international help it can rely on. The MIC plays three important roles during emergencies [56]. These are the following :

- *Communications hub.* The MIC is an important point for the exchange of requests and offers of assistance. This fact helps to reduce the administrative tasks of the 30 participating states' in connecting with the affected country. It provides a control forum for participating states to share information about the available resources.
- *Information provision.* The MIC disseminates information on civil protection preparedness and response to participating states as well as to interested persons. As part of this role, the MIC disseminates early warning alerts on natural disasters and circulates the latest updates on ongoing emergencies and Mechanism interventions.
- *Supports coordination.* The MIC facilitates the provision of European assistance through the Mechanism. This takes place at two levels: at headquarters level, by matching offers to needs, identifying gaps in aid and searching for solutions, and facilitating the share of common resources where possible; and on the site of the disaster through the appointment of EU field experts, when they are required.

Inside the European Union, the Community Mechanism can be activated through the MIC by any state that is participating and who looks for quick international assistance after a major disaster has happened. A state usually calls on the Mechanism when the effects of the disaster cannot be matched by its own civil protection resources. As soon as the MIC receives a request for assistance, the Centre immediately forwards it to its 24-hour network of national contact points. The contact points represent the participating states' civil protection authorities. They check their available resources and inform the MIC whether or not they are able to help. The MIC then matches the offers made to the needs and informs the requesting state of the type and quantity of available assistance from the Community. The arrangements for the dispatch of the accepted assistance (delivery, transport, visa requirements, customs, etc.) are performed in a direct way between the offering and requesting states. If it is required, the MIC may play a facilitating role. Any intervention teams or assistance sent from the EU to a disaster area remains under the direction of the national authorities of the affected country, which has the right to ask European teams to stand down at any time. European teams refer to local law and should perform their tasks in conformity with national rules.

- ***Common Emergency and Information System (CECIS)*** The Common Emergency and Information System (CECIS) is a web-based alert and notification application which has the intention to facilitate emergency communication among the participating countries. It provides an integrated platform to send and receive example alerts [55]. CECIS facilitates the communication between the MIC and National Authorities, making response to disasters faster and more effective. It therefore aims to better protect citizens from natural and technological hazards. This interconnection will facilitate the exchange of information and experience between authorities responsible for Civil Protection in order to improve the capabilities of these organizations to deal with the different phases of emergencies. Its main task is to host a database on potentially available assets for assistance, to handle requests for assistance on the basis of these data, to exchange information and to document all action and message traffic. The end-users of CECIS are the MIC and the National Contact points of the Member States. Through the CECIS, operational information can be exchanged in a secure and reliable way, as needed for the effective implementation of the mechanism.

- ***Training Program*** The training program has the intention to improve the coordination of emergency management assistance interventions. Its task is to ensure the compatibility between the intervention teams of the participating countries. In order to prepare as best as possible to one's response to natural or manmade disasters, the Community Mechanism for Civil Protection foresaw three types of measures: training courses, simulation exercises and exchange of experts. These aim at improving personal response competencies and ensure compatibility between the intervention teams from the different European countries.

To conclude, it is necessary to underline a further European Tool for Emergency Management not dealing with the Community Mechanism, named ***Global Disaster Alert and Coordination System (GDACS)***. The Global Disaster Alert and Coordination System provides near real-time alerts about natural disasters around the world and tools to facilitate response coordination, including media monitoring, map catalogues and virtual on-site Operations Coordination Centre. GDACS provides a platform allowing stakeholders in international disaster response to exchange disaster-related information in a structured and predictable manner, particularly in the response phase of disasters. It is not aimed at informing the potentially endangered population. The GDACS combines existing disaster information management systems. Early information is expected to be uncertain and will be refined as better information becomes available. GDACS also establishes an awareness-building process among its stakeholders and promote capacity building in disaster-affected countries in order to facilitate their participation in the system. The Community's ability to respond immediately to natural disasters depends on the availability of early warning systems enabling the Member States and the mon-

itoring and information centre to take the necessary action within shortest time possible.

3.3.2 Emergency Management at National Level in Europe

In this section are introduced the emergency management structures of different European countries[1]. In general, the emergency management depends on the legislation of the respective country. Basically, in most European countries the structures are highly similar, namely always organized as a hierarchy of several levels, which for WORKPAD means that these hierarchical structures need to be addressed and flexibility for the concrete instantiation of a WORKPAD system in one country need to be provided. In most European countries, emergency management is a task assigned to one single institution or few public structures. This section presents the emergency management structures of Austria, Germany, France, Spain, Czech Republic, and the United Kingdom. Taking into account the development over the last years, the European Commission have been investing heavily in new technologies and methodologies in order to improve the State-of-the-Art in disaster management. Many projects have been financed in this field until now. But what it still lacks is the availability of a common European infrastructure [8].

Austria

In Austria the emergency management is regulated by law of the nine provinces. In each of the nine provinces of Austria an Alarm Center is installed. The Provincial Alarm Centers coordinate the public safety organizations during major disasters which affect more than one region. If a disaster hits one region then the regional administrative authority is responsible for coordinating the different organizations. In contrast to other countries, Austria has no special emergency management unit. Emergency management in Austria is provided by the voluntary public safety organizations, which are motivated, well-trained and optimally equipped. Furthermore Austria has installed a federal alarm center on the national level. The task of this center is to recognize hazardous situations, to give out warnings and alerts to the public, to coordinate tasks in disaster prevention and to become active within nation-wide and international disaster response activities.

Germany

In Germany the emergency management in peacetime is a matter of the "Länder" (i.e. regions). Within the Länder the local or regional authorities are responsible for assistance during an emergency event. The main institutions available to the local and regional authorities to provide assistance are the fire services (professional and voluntary fire brigades). The fire services also get support from other public

[1]The selection of countries was done randomly by particularly considering the WORKPAD partners' country of origin.

safety organizations like the Red Cross in case of medical services. The Federal Government complements the resources of the Länder with the GMLZ (Dispatch and site information center) and deNIS (Disaster Preparedness Information System of Germany) in case of major disasters. The Federal Institution for Technical Assistance (THW) is for rescue services which can be requested from the Länder authorities for the assistance on emergency management activities.

France

The Directorate of Public Safety (DSC), which is attached to the Ministry of the Interior, manages the national emergency service in France. The DSC coordinates the action of the local rescue services responsible for aid operations. Its operational center (CODISC) ensures round the clock monitoring of large-scale rescue operations at a national level and abroad. In addition to the CODISC six inter-regional centers of operational coordination on public safety (CIRCOSC) are established in France. They ensure the coordination of the aid and rescue operations under the authority of the prefect of the inter-regional area.

Spain

In Spain the General Directorate for Civil Protection, which depends on the Ministry of the Interior, is the national body working out the emergency intervention programs. The Spanish system is based on preliminary planning and cooperation between those who have at their disposal the various resources which can be implemented to cope with emergencies. The system is decentralized and allows widespread intervention of all the resources of the country in order to cope with an emergency. The Basic Standard lays down the requirements for emergency management plans. It sets out the criteria for coordination between the plans of various administrations (central, autonomous, local) and the general framework for developing the competencies of these administrations. Each administration can organize and manage its emergency management systems with complete autonomy, but must respect the principles of inter-territorial complementarity, subsidiarity and solidarity.

Czech Republic

The General Directorate of the Czech Fire Rescue Service (CFRS) secures execution of public administration in emergency management that is enacted in organizational structure of the Ministry of Interior. CFRS coordinates activities in the area of emergency management with central and other bodies of public administration. The Regional Fire Rescue Services are state organizational bodies to ensure the activities connecting with the execution of state administration in the matter of Civil Protection. They fulfil tasks connected especially with the preparation for extraordinary events, providing of rescue and liquidation work and protection of

citizens. Bodies of the municipality ensure readiness of the municipalities for extraordinary events. They participate in providing rescue and liquidation works and protection of population.

United Kingdom

It is fundamental to the arrangements for dealing with disasters in the United Kingdom that the first response happens at the local level. In the event of a disaster, as far as the immediate reactions are concerned, reliance is placed on the emergency plans made by the emergency services (police, fire and ambulance), local government, public and health services, those responsible for industrial installations and others including the voluntary sector. The preparations of these plans - which are based on the guidance and instructions given by government or other services - are often coordinated by local authorities, although it is the police who coordinate the response. If a disaster reaches a scale which the resources that are available on the spot cannot manage, supplementary resources might be called from neighboring authorities and organizations as well as from central government. Only an exceptionally massive disaster would justify coordination at (central) government level. Additional resources might also be requested, if necessary, from neighboring countries, from member States of the EC or from the NATO.

3.3.3 Conclusions

After the analysis of European legislation and the situation of emergency management on national level it can be drawn the following conclusions regarding the different aspects that shall be considered in WORKPAD:

- ***Coordination.*** The prerequisite of a comprehensive coordination on a European level is represented by improved coordination at the Community level. In order to provide effective coordination, the WORKPAD system must respect the emergency structures existing in European countries. This must be accomplished by flexible system architecture that can be adapted accordingly. Furthermore, WORKPAD must allow intra-national communication and communication between on-site emergency teams to and between different organizations that control the emergency activities from a Back-End control room.

- ***Link to MIC.*** WORKPAD on an organizational basis should foresee to integrate the expertise of MIC. and talk about cooperation possibilities especially with regard to how MIC and WORKPAD could be best connected referring to coordination aspects in emergency management within the European Union.

- ***Interface to CECIS.*** WORKPAD should provide an interface to the CECIS system in order to have an additional source of alerts of emergency incidents.

WORKPAD has the focus on the response phase (phase that follows after the disaster has happened) and especially in this phase it is important to quickly response to alerts, and hence to be alerted as soon as possible. Therefore it is of advantage for WORKPAD to be connected with the CECIS system.

- ***Interface to GDACS.*** WORKPAD should provide an interface to the GDACS system in order to get the latest information concerning real-time alerts on a world basis and to have access to tools to facilitate response coordination. As WORKPAD should use relevant existing systems, a connection to the GDACS system should be available.

- ***International Charter.*** WORKPAD operators should be recognised as "test authorized user" of the International Charter in order to more effectively exploit the benefits of this institution.

- ***Emergency Management at the national level.*** Concerning emergency management, most of the structures and processes are regulated on a national level. The EU merely gives advices and provides recommendations, or tools respectively (such as MIC or CECIS). This means for the WORKPAD project that these national regulations must be taken into account when designing the system. Fortunately, in most European countries the structures are highly similar, namely always organized as a hierarchy of several levels. Nevertheless, to address the differences, the WORKPAD system must be able to cope with the potentially different hierarchical structures and must provide the according flexibility for the concrete instantiations of a WORKPAD system for a specific country. A certain level of generalization is required in order to support the different occurrences of emergency management systems of the European nations.

These conclusions drawn from examination of European initiatives and emergency management on national levels are considered in the definition of the WORKPAD requirements (see Chapter 5). In the model used to describe requirements it is defined a specific field depicting the source of one particular requirement. Two types of sources are defined: (1) user analysis (i.e., the Calabrian case study) denoted by 'U' and (2) investigations of related work and/or EU regulations denoted by 'I'. These conclusions here are reflected in requirements classified as having an 'I' source. The comprehensive listing of the WORKPAD requirements can be found in Section 5.5.

3.4 Related European Projects

This section summarizes the results of the analysis of some former European Projects which cover similar topics like WORKPAD either from an application area (i.e., emergency management) or from an architectural point of view. These are investigated regarding their applicability to the WORKPAD context, and the

analysis concerns not only their relevant concepts or components, but specially the requirements elicitation process of each project, the outcomes, and how these can be re-used or exploited into WORKPAD.

The real analysis were performed working on a set of 14 European projects; in this thesis, for the sake of brevity, it is shown only a relevant subset of the projects that were subject to investigation.

In particular, the following projects have been analysed:

AMIRA/RIMSAT	OASIS/EGERIS
ORCHESTRA	LIAISON
WIN	

Table 3.1 gives a high-level overview description about the adopted architecture approach and summarizes the main outcomes according to these criteria:

- Does the investigated project cover Front-End or Back-End issues or both?
- What is the architectural approach?
- Does it support mobility?
- Which application area is addressed?
- And what are relevant system architecture aspects?

In a first step, the presented projects were checked via the project website for the essential documents which describe the requirements analysis methodology deployed. In cases where the relevant documents were not available the project coordinators were contacted by e-mail. In a second step, the user requirements were evaluated with respect to WORKPAD, if appropriate listings were available and applicable to our projects context.

In the following sections, these projects are presented where the documents related to requirements were available and accessible. After a short description about the general objective of the project, the methodology of the requirement analysis is illustrated. At the end of each project it is discussed the relevancy with respect to our requirements elicitation methodology and potentially valuable requirements for WORKPAD.

3.4.1 AMIRA Project

The technical challenge that project AMIRA[2] faces is to develop a set of Back-End reusable components using search, reasoning, speech dialogue technology and collaborative working techniques that can be used to create a variety of applications

[2]http://www.amira.no/

Project	Covering FE/BE/Both	Architectural Approach	Mobility Support[a]	Application Area	Relevant System Architecture Aspects
AMIRA/RIMSAT	FE	Agent-based Client/Server	Yes	Training for roadside or fire assistance	Interface to knowledge sources, workflow management, ontology mapping
LIAISON	Both	Client/Server	Yes	Emergency management and other business applications	Location server and LBS platform
OASIS/EGERIS	BE	SOA	Yes	Emergency management	Data integration, ontology engineering, interfacing heterogeneous (legacy) systems
ORCHESTRA	BE	SOA	Yes	Risk Management	Semantic interoperability, BE data integration, spatial service
WIN	Both	SOA (WebServices)	Yes	Risk management	Service orchestration, user profile management and cataloguing services

Table 3.1: Summary of Analysis of Relevant European Projects

[a]"Mobility Support" means if mobile operators on-field can be supported by the (intended) system.

for mobile workers operating in safety or business critical situations in the field. The AMIRA project is the follow-up project of the RIMSAT[3] project, which was funded in the 5th Framework Program (IST-2000-28655).

Requirements Analysis Methodology

The socio-economic study of user needs [11] addresses the needs of mobile workers in terms of computerised support when they are working in the field, as well as the type of support that their employer organizations are prepared to provide. This study served as the basis for the requirements analysis. The two areas of applications are emergency fire services and vehicle roadside assistance. The methodology which is used to acquire the requirements is divided in three analysis levels:

1. Knowledge base interrogation and analysis.
2. Interviews and questionnaires.
3. Requirement assessment.

In the first step existing knowledge bases and documents were analysed to gain a short overview of the current state of working, information sources and end-user skills. The second level involves both the review of the first level results and the acquisition of knowledge about end-user domains. In the interviews and questionnaires the results of the first level are checked and new data is acquired, which does not exist in databases. Comparable to the first level, the objectives of the second level are to elaborate the current state of working, the information sources used and the skills of end-users. The requirement assessment represents the conclusions drawn from the previous levels. At this level working processes are identified that frequently occur in the fire service domain. These working processes are the basis for elaborating user scenarios. Moreover, user needs are elicited, formulated, defined and evaluated based on the first and second level. The analysis of the user needs cover the development of the user requirements and the use cases.

AMIRA's Relevancy for WORKPAD

The socio-economic study of AMIRA provides the guidelines for the development of the AMIRA system. In this study the requirements for the fire service domain are very general and described in a short list. For a better understanding requirements were depicted in form of UML Use Case diagrams and practical examples were given. For WORKPAD, the vehicle roadside assistance scenario is not relevant, so we focused on the emergency fire services scenarios. The main outcomes of the AMIRA requirements analysis relevant for WORKPAD are:

- The mobility of mobile operators must be ensured. The system must not hinder mobility and service must be provided at any location.

[3] http://www.rimsat.com/

- The system's user interfaces must be designed according to usable in the specific context of the mobile emergency operators, which are under stress, wearing gloves, helmets etc.
- The large number of different information sources has to be integrated into one platform.
- Mobile data access has to be provided.
- The information system has to be compliant with the workflow and processes of the mobile operator in the field which is highly dynamic. Hence, added-value can only be provided if their work can be supported by adaptive systems.
- Collaborative processes in the field must be improved or facilitated at all by providing appropriate knowledge sharing or exchange mechanisms.
- A closer collaboration between the control rooms and their mobile operators in the field must be offered.

3.4.2 LIAISON Project

The aim of the LIAISON[4] project is to provide Location Based Services (LBS) for a wide range of mobile workers by combining existing standards and techniques. Among the scenarios for validation are fire brigade intervention and incident management.

Requirements Analysis Methodology

In the Service Definition Document [13] the methodology for the user requirements analysis is described. In LIASION, two methodological concepts are used. The SPIN approach was developed by Neil Rackham [43] for the purpose of customer-oriented selling. Today this approach is used in research and development projects as a guideline for structured user interviews. As indicated by the acronym the method comprises four steps: (1) Situation, (2) Problem, (3) Implication and (4) Need. In each step different questions are asked do get a better overview of the steps mentioned here. The information which is gathered in the interviews from the SPIN approach flows into the use case methodology. This method is used for describing processes in different situations including business processes or functional system requirements descriptions.

LIAISON's Relevancy for WORKPAD

The Service Definition Document [13] provides the first definition of services of the LIAISON project and is intended to be the expression of user needs and requirements defined in non-technical terms. The Service Definition is the basis for

[4] http://liaison.newapplication.it/liaison/

the further definition of mission and technical requirements. The mission requirements are documented in the Mission Requirements Document [12]. The document for the technical requirements is not available. In the Mission Requirements Document the high-level system's functional and operational requirements as well as the non-functional requirements of the product vision of the six test-cases are presented. The two relevant test cases are fire brigade and incident management. In the fire brigade scenario many hardware related requirements with respect to weight of device, power time, and resistance are discussed which are beyond the scope of WORKPAD. Nevertheless, requirements mentioned and interpreted are:

- A reliability of 100% is required. If the system for some reason is not available it must notify the relevant parties early enough.
- The additional device is only accepted if it provides an added-value and integrates seamlessly with the systems (hardware and software) present.
- Devices must be unobtrusive, ideally integrated into clothes.
- Mission-critical information must be delivered to responsible persons, such as location of team members potentially in danger.
- To secure data transmission, authentication and secure communication shall be provided.

Relevant requirements of the incident management scenarios are:

- Only authorized agencies shall be allowed to retrieve data.
- In order to preserve network resources, "data light" methods of communication shall be used.
- Data transmission between emergency service device and Service Access Points (i.e., Back-End) shall be bidirectional.
- Transmitted data shall be accordingly visualized on the Front-End devices.
- Human-Machine Interface regulations shall be considered when designing the user interface for devices in emergency situations.
- The system shall automatically identify the location of field operators.
- The level of detail of geographic information shall be user-definable.
- Front-End devices shall be capable of storing and retrieving data to and from a local memory.
- The system shall be integrated with diverse (also external) data sources.
- Front-End devices shall be able to capture, send, receive, and visualise different information such as photographs.

3.4.3 OASIS Project

The OASIS[5] project is the follow-up of the EGERIS[6] project, which was funded in the 5th Framework Program (IST-2000-28345). The aim of the OASIS project is to define and develop a first version of an open, modular and generic disaster and emergency management system in order to improve the effectiveness and efficiency of all agencies within the European Union who are likely to be involved in the management of disaster and emergency operations.

Requirements Analysis Methodology

In the OASIS project a two-stage methodology was used [14]. In the first stage, structured interviews were taken with different emergency responder organizations within each of the nations represented in the consortium. The user requirements assimilated from the interviews reflect the current operational needs of the emergency responders. In addition to the interviews, publicly available material was searched for relevant user experiences which were relevant to OASIS. In a number of cases National Government doctrinal documents were used to compile specific requirements. In parallel, a synthesis of user requirements from other projects such as EGERIS was undertaken, building up on previous work. From all these documents and interviews common and generic requirements were extracted. In the terminology of OASIS "generic requirements" are requirements which are common to different organizations. In the second stage, a second round of interviews took place which collected information from users in other countries. This process completes the common outcomes of the national disaster management analysis processes. The final synthesis of the user requirements took into account the first and second round of interviews, information gathered from publicly available material and input from the strategy group of the project.

OASIS' Relevancy for WORKPAD

In general the context of OASIS is very similar to the one of WORKPAD, although the focus is different. Hence, it is a highly valuable source of input for WORKPAD. The OASIS user requirements are extracted from the reports of the above described processes. The requirements are kept on a generic level to be valid for the different addressed users. The user requirements were assigned to nine different classes. During the requirements process four high-level key issues were identified by the OASIS consortium:

- Interoperability regarding sharing information.
- The problem of different communication networks.
- The used applications are still very basic.

[5] http://www.oasis-fp6.org/
[6] http://www.egeris.org/

- Need for advanced decision support in emergency management.

On a more detailed basis, interesting requirements for WORKPAD can be summarised as follows:

- An emergency system such as the one intended by OASIS, which shares some similarities with WORKPAD, shall be scalable, modular, flexible, portable and interoperable.
- The system must work in and address the specific characteristics of collaborative environments.
- Several components (such as Front-End applications) shall also work in a stand-alone mode.
- Open standards shall be used for communication, information storage and exchange, collaboration and security.
- User-friendly interfaces shall be provided.
- The used equipment shall be robust and energy-efficient.
- The system shall provide knowledge-based tools to support workflows and plan resources.
- Maps and geo-referenced data shall be integrated into the system in order to support operators.
- Users of the system shall be able to store, retrieve, filter and visualise all the information relevant to an emergency incident.
- Individual operators in the field shall be addressable and reachable.
- The system shall provide access to diverse data sources (e.g., meteorological and environmental information).
- Field operators shall be equipped with portable devices which can be used to exchange information such as images and communicate with other operators and the Back-End.
- Different levels of authorisation shall be definable.
- The system shall be able to process and present "situation awareness" information such as geographic extent, number and type of casualties, extent of damage, date and time information, level of importance, level of confidence, and level of severity.
- Seamless communication between system users must be provided. The underlying network infrastructure shall be redundant (avoidance of single-points-of-failures) and resilient and thus offer a high availability rate.

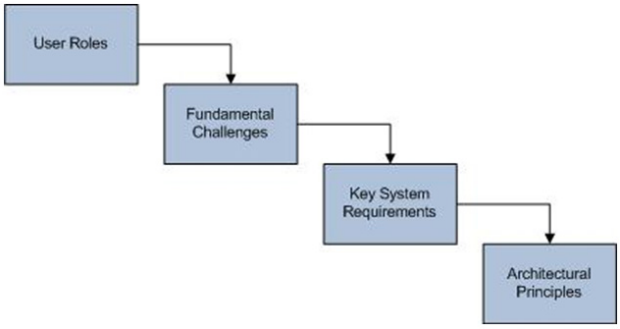


Figure 3.7: Requirement analysis in ORCHESTRA

- In the event of a network failure, graceful degradation shall be applied.
- Field operators shall have the possibility to search and retrieve information regarding resources and materials.
- Sensitive information shall be secured by encryption.
- The system shall offer possibilities to priorities message to provide guaranteed delivery according to defined priority levels.
- The necessary infrastructure on the emergency site shall be installable within half an hour.
- Decision makers shall be supported in assessing and managing risks.
- Adaptive workflow management shall assist in dynamic re-planning of actions by evaluating the current and alternative courses of action.

3.4.4 ORCHESTRA Project

The objective of ORCHESTRA[7] project is to design and implement an open service-oriented software architecture that will improve the interoperability among actors involved in disaster and emergency management operations.

Requirements Analysis Methodology

From the ORCHESTRA project, a document of requirements of the ORCHESTRA Architecture and the ORCHESTRA Service Networks is available [15]. A so called "line of arguments" (see also Figure 3.7) is used to define the requirements. The line of arguments starts by describing the different types of users of the system and their roles. These user roles are connected with fundamental challenges which are considered relevant to the system. The fundamental challenges lead to key systems requirements and finally to architectural principles. Fundamental challenges

[7] http://www.eu-orchestra.org/

are those important sets of issues which the architecture of ORCHESTRA has to manage. In the requirements document the user roles, fundamental challenges, key system requirements and architectural principles are identified and the relationships to one another are described. The description of each element is done by a table that indicates the dependencies of the element to the related elements of the previous and subsequent step in the line of argument. The reason for this type of argumentation chain is to provide a foundation for the architectural decisions.

ORCHESTRA'S Relevancy for WORKPAD

The requirement analysis of the ORCHESTRA project is cut into four chains as described above. For this analysis, only the "key system requirements" chain is presented where the others can be found in the "ORCHESTRA Architecture and the ORCHESTRA Service Networks" document [15]. The fundamental challenges defined within the ORCHESTRA project (i.e., scale and scope, integration/collaboration, long lifetime, quality, transparency, access control) lead to the following key system requirements:

- Openness: The architectural specifications of ORCHESTRA are multi-vendor, public available and free of charge.
- Scalability: The system has to scalable in terms of number of autonomous systems, number of concurrent users, number of collaborating services and number and size of data sources.
- Usability: Usability facilitates the user's access to the system. The usability of the system is categorised according to the different user's expectations and needs.
- Accountability: The elements and functions of the ORCHESTRA Service Network should be accountable for their characteristics and behaviors.

On a more detailed basis, relevant and interpreted requirements for WORKPAD can be summarised as follows:

- The main components of the system must be designed such that they interact (internally and externally) via open interfaces.
- The system must be open to facilitate integration and collaboration across organisational structures, technologies, data sources, domains and semantics.
- The architecture must be designed in a technology independent manner, to produce a generic infrastructure that addresses openness.
- The system must address scalability by accommodating future increases in size (e.g., number of concurrent users, integrated autonomous sub-systems, collaborating services, data sources).

- To provide accountability, the system should be able to report to or modification of data or services.
- The system should ensure that only authorized access is granted.

3.4.5 WIN Project

The objective of the WIN[8] project is to design an information architecture ("info-structure") based on State-of-the-Art information technologies, protocols, and standards. At the same time interoperability with existing risk management services shall be provided. For these purposes WIN will investigate multiple stakeholder business and organizational models appropriate for the future deployment of the info-structure. WIN covers the thematic domains of oil spill pollution, forest fires and floods.

Requirements Analysis Methodology

The WIN User Requirements Specifications [16] presents the detailed user, data and service provider requirements. This document is used for the specification of the WIN architecture design in terms of data model and generic WIN services. The document covers several domains according to a multi-risk approach to specify concrete requirements on the one hand. On the other hand, however, also generic requirements applicable to other risk domains beyond the scope of WIN. In this project a two-stage methodology for the requirement analysis was used. In the first stage the requirements collection process was performed by the thematic actors of the project which already have knowledge because of other projects in term of disaster management. In the second stage additional requirements resulting from French emergency management lessons learned on daily work and expertise were integrated. During the requirements capture and analysis process, a user oriented classification approach has been adopted because this classification represented a good model for the scenarios. The classes are "Risk phases and operational activities", "Users groups" and "Types of faced risks".

WIN'S Relevancy for WORKPAD

The user requirements have been organized according to a defined structure which alleviated the subsequent transformation into technical specifications for the system architecture. The requirements are classified thematically and sorted by technical attributes such as security, communication, service performances, products, data management and system operational.

The following interpreted requirements are considered in WORKPAD, too:

- Scalable communication services shall be provided according to the needs of a particular data transmission.

[8] http://www.win-eu.org/

- Mobile communication shall enable the movement to and within the affected site and to establish a connection between field operators and the control rooms.
- The system shall integrate communication within and between the involved organizations.
- Applications shall be developed that support or facilitate the collaboration between operators by sharing (multimedia) data to allow real-time interactive team work.
- Satellite communication shall always be available as a back-up during a disaster.
- Interfaces to the Infrastructure for Spatial Information in Europe (INSPIRE) shall be provided to complementary enrich the available amount of data.
- A flexible and distributed architecture shall be provided to share data between different actors.
- Data export and import mechanisms shall support common GIS formats.
- A semantic search engine shall be provided.
- The information coming from different geographic sources must be portrayed in a standardized way.
- Common standards and definitions must be used if data sharing and common systems are to be established.
- The system should foresee and distinguish different authorisation levels.
- Quality of service of data transmission shall be configurable.
- The system shall integrate existing communication norms (e.g., TETRA, UHF/VHF).
- The system must be easy to use.

3.4.6 Conclusions

To conclude, several issues seem to be crucial for IT-based emergency system, such as integration of various and heterogeneous data sources, Back-End and Front-End communication, context-sensitive adaptiveness, workflow-management, flexible interconnection of Front-End team members, and user-oriented software services for on-site operators. WORKPAD touches all of these issues. The unique selling proposition of WORKPAD is that it combines Front-End and Back-End issues and treats them equally important. The one component delivers added-value

	Review of former projects	Review of publicly available documents	Interviews	Second interview iteration
AMIRA	(x)*	x	x	–
LIAISON	–	–	x	–
OASIS	x	X	x	x
ORCHESTRA**	n.I.	n.I.	n.I.	n.I.
WIN	x	–	x	x

x ... This method was used
– ... This method was not used
n.I ... No information available

*) Only the forerunner project RIMSAT was reviewed
**) In the requirements document of the ORCHESTRA project only the theoretical methodology is described. It contains no information about the information retrieval process from the users.

Figure 3.8: Comparison of the User Methodology of Related EU-Projects

to the other one. Nevertheless, both are designed in a flexible way such that extensibility, exchangeability and integration of external components are possible. For instance, the WORKPAD Back-End data integration sub-system shall be able to integrate other systems providing emergency management relevant information which may be compliant to other standards or specifications. Moreover, with respect to Front-End applications, WORKPAD provides a framework that allows extensibility with new services that can either exploit the WORKPAD infrastructure available through the defined interfaces or can keep on using its legacy infrastructure, or both. This legacy infrastructure, in turn, can also be integrated into the WORKPAD emergency management system. This flexibility with respect to the Back-End and the Front-End and the interconnection between these two subsystems results in the added-value that shall be offered to emergency organizations and operators.

Apparently, all investigated projects adopted similar methods which can be summarized as review of related work and interviews. Figure 3.8 confronts the analysed projects and the deployed requirements capturing methods which should serve as a summary of the descriptions given in this section.

Notably, in LIAISON according to the methodology description no review of former projects of public documents was conducted. In the description of the ORCHESTRA project the theoretical "line of arguments" approach was discussed but no details were given about any user involvement. The other projects all involved user interviews and in OASIS and WIN also a second iteration of interviews was conducted. On a side note, according to the examinations, the project most relevant and most close to WORKPAD is OASIS. The context and objectives of this project most closely match the ones of WORKPAD. Furthermore, the requirements elicitation process is also very similar; moreover, the relevant requirements list extracted from OASIS is the most comprehensive one and is accordingly taken into account.

From the analysis it is also deduced that in WORKPAD, besides the theoretical examination of related work, it is necessary to work together as closely as possible with real users in the practical work (the case study of emergency management in

the region of Calabria). As the result of the analysis was that in former projects it proved beneficial to have a second iteration of interviews, the requirements analysis is designed with respect to the show case accordingly. We also decided not to use interviews as the only human-computer interaction technique but also to deploy further ones, such as scenarios, focus group meetings, storyboards, and task analysis (see Chapter 4) and to use them iteratively.

Moreover is possible to see that many of them also in the final version were relatively high level. Some of the more specific ones are also perfectly valid for the WORKPAD system. The remaining requirements have to be adapted to the specific context of the WORKPAD project which on the one hand is defined by the scope and objectives of the project but on the other hand by the outcomes of the specific case study of the region of Calabria. This results from the fact that it is adopted a requirements elicitation dualism; i.e., (1) *bottom-up* by the practical case study generating very concrete requirements which have to be slightly generalized and (2) *top-down* by theoretical examination of the related work resulting in rather abstract requirements which must be broken down to make them more appropriate for the concrete example of WORKPAD. To conclude, a synthesis and a summary of the most relevant requirements are provided in the following listing:

- ***Process support.*** The specific emergency management structures and involved workflows must be supported. Due to frequently changing environments, these processes are highly dynamic which makes adaptive process management necessary. Such process management should also increase the current level of automation.

- ***System design.*** In order to provide greatest possible availability, the system must be designed in a decentralized and redundant manner. The frequent joining and leaving of network nodes must not affect the scalability of the system in a negative way.

- ***Data integration.*** The large number of different information sources must be integrated into one platform. Especially, geo-referenced data (such as digital maps) shall be integrated into the system in order to support operators.

- ***Communication.*** More appropriate and reliable communication means within and between involved organizations and between field operators and control rooms must be provided. Speech as well as data services shall be delivered to the Front-End operators. Network quality-of-service shall be configurable to exploit the network and to provide ideal data delivery.

- ***Field operators.*** Collaborative processes in the field must be improved. Front-End devices shall be able to capture, send, receive, and visualize different types of information. The mobility of mobile operators, however, must be ensured and never hindered.

- ***Security.*** Mission-critical data is provided and transferred over (wireless) networks. To secure data transmission, authentication and secure communication shall be provided.

- ***Usability.*** The system's user interfaces must be designed according to be usable in the specific context of the mobile emergency operators. Unobtrusiveness of the service shall be guaranteed. The operator must not be hindered in his activities under no circumstances.

This synthesis represents merely a summary of the analyzed requirements which we subsequently took into account when deriving the WORKPAD requirements. As already described in the conclusions of Section 3.4, our requirements description model defines the source field where investigations of related work and/or EU regulations are denoted by 'I'. These conclusions here are reflected in requirements classified as having an 'I' source and are listed in Section 5.5.

Chapter 4

The Adopted Methodology

The environment in emergency or disaster management scenarios is very critical for many aspects as a small mistake can put many human lives in danger. So, a formal methodology and well-designed principles, for designing and developing an interactive system for supporting emergency operators, plays a critical role in the success of the resulting system. In the WORKPAD project, the potential users have a significant impact on the design and development process and are considered to be the most important drivers of innovation within the project. The users must to be actively involved in all stages of the system engineering process. The purpose is not just the simplicity but to develop a working system with functionalities and capabilities with an adequate level of usability.

With respect to other methodologies used in previous research projects about emergency management (see also Section 3.4), the main contribution of the work held in the WORKPAD project is a careful use of all possible techniques to get feedbacks from the users. This requires a continuous contact with real end users, by leading them not only to answer to simple questions but also to think about their suggestions and impressions.

This Chapter aims to detail the adopted methodology and the techniques used to design and to make the WORKPAD system more interactive and more efficient in emergency scenarios, according to the User-Centered design (UCD) approach [4].

The Chapter is organized as follow :

- Section 4.1 describes the overall adopted methodology to design the system.
- Section 4.2 defines a high-level conceptual framework for the definition of the WORKPAD requirements, including a description of all the different techniques performed to get feedback from the users (interviews, scenarios, task analysis etc.).
- Section 4.3 presents and describes the evaluation and validation plan performed in WORKPAD, including a description of all the different test types

used to assess the project outcomes (mock-up tests, cooperative evaluations, controlled experiments, etc.).

4.1 The User-Centered Approach

The approaches and technologies developed by the WORKPAD project have been radically different from the existing solutions and there was a need for a certain amount of experimentation. Thus the solutions developed within the project ***needed to interactively be explored with the users*** and the new ideas and concepts needed to be tested.

The users are able to offer concrete requirements, but due to the innovative approaches proposed, further requirements needed to be explored with the use of prototypes as a requirements elicitation tool. Therefore prototypes have been used to present ideas and alternative approaches and to get rapid feedback in order to provide direction for the evolution of the WORKPAD system and its specification. A standard *Waterfall model* (for further details see Section 3.1.1) of the project life-cycle would have been counterproductive and the best fit was provided by a *Spiral life-cycle* [9]. As already detailed in Section 3.1.4, the spiral life-cycle supports an iterative evolution process and it was therefore the one adopted by the project.

The spiral life-cycle is iterative, starting with initial requirements gathering followed by rapid proposition on new basic research ideas, design and development of a prototype implementing them, that is then tested and validated by the users and is enhanced in the next iteration with user feedback and additional requirements. The test results provide a new set of requirements, through redefining or completing the old set. With the second set of requirements a new design and development is performed. After that, users are required to test the system again. That process is continued till a complete set of requirements that meets all user requirements has been implemented. A summary of this approach has already been shown in Figure 3.3.

The Prototyping approach (for further details see Section 3.1.2) has been suitable for our conditions, because users are able to check the system in the early stages. From the first prototype users are able to assure the capabilities of the system, and following stages let them confirm that their needs have been met. This allows a periodic re-validation of the project development by the users group. In each iteration of the cycle, project user acceptance is required and partial results are disseminated among all of them. In the WORKPAD project, we have realized 2 cycles (a first prototype during the 2nd year, validated and revised before the final outcome of the 3rd year).

In order to realize the full potential of the approach proposed within the WORKPAD project, we have payed special attention to the involvement of the potential users in the process of research, development and prototype creation and introduction. The (potential) users of the WORKPAD project have a significant impact on the development and creation process and are considered to be the most

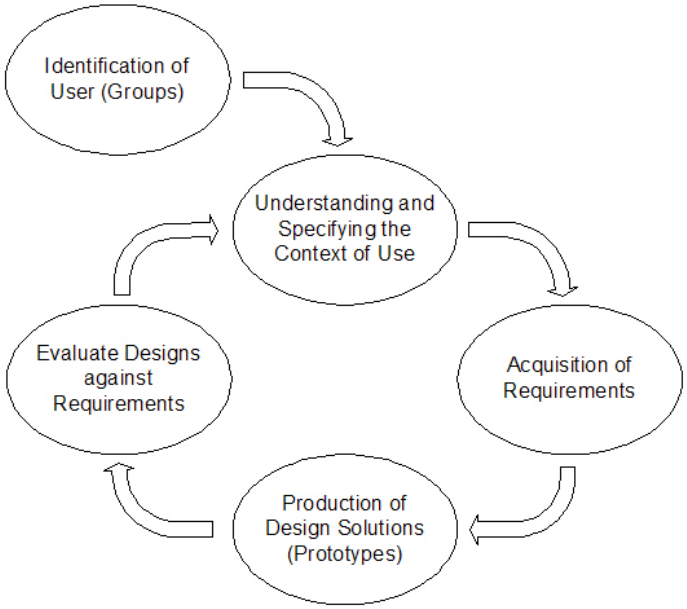


Figure 4.1: The main design activities of the WORKPAD project

important drivers of the innovation within the project. They were actively involved in all stages of the application creation. The WORKPAD prototype was developed by using a user-centered and participatory approach, adapted from the international ISO 13407 standard [1], that recommend to develop an iterative and incremental approach. User-Centered approach places the person (as opposed to the "thing") at the center; it is a process that focuses on cognitive factors (such as perception, memory, learning, problem-solving, etc.) as they come into play during peoples' interactions with things and/or systems. It seeks to answer questions about users and their tasks and goals, and further on uses the findings to drive development and design. User-Centered Design is increasingly seen as essential for the creation of successful products and prototype ICT (Information and Communications Technology) applications; it enables understanding of the needs of users early in the design and development process, shaping product design to their needs, and validating the acceptability and usability of the ICT product or prototype application.

Figure 4.1 shows the general steps that have been undertaken in order to reach the specified objectives of the User-Centered approach within the WORKPAD project: identification of different user groups, user scenario description, definition of the user and system requirements, production of prototypes and usability testing. The general guidelines about how to use this kind of approach are detailed in Section 3.2.1.

Section 4.2 describes the methodology exploited to perform the first three steps of the adopted approach :

- *Identification of different user groups*. In the first stage we have identified users groups, design questionnaires (aimed at collecting data related to the needs and expectations of users, and evaluating design alternatives) and execute interviews with the selected user groups.

- *User scenario description*. Then we have defined two main user scenarios, specifying how users carry out their tasks in a certain context. They provide examples of usage as an input to the design of the application, and serve as a basis for subsequent usability testing. Moreover, starting from scenarios, we have derived some storyboards and performed a Task Analysis, in order to understand in deeply the atomic steps that a user fulfills in facing a real situation.

- *Definition of the user and system requirements*. The "knowledge" obtained by the two previous point served as the basis to derive the user requirements. They have been analysed from the point of view of the potential end users and in detail described with UML use cases. System requirements were derived from the User requirements and from the knowledge of technology and constraints.

Once the requirements were defined, each partners participating to WORKPAD project started to design *Prototypes* for its own component. Notably, this Thesis will give details about the mock-ups and prototypes realized for the Process Management System (PMS), one of the main component of the system. Chapter 7 is dedicated to this topic.

In order to ensure a complete user-driven development of the WORKPAD system, the last stage of our approach involves *Usability Testing*. Section 4.3 describes the evaluation and validation methodology we performed, necessary to carry out an effective assessment of the project outcomes.

4.2 Requirements Engineering Activities

In the WORKPAD project, twofold (bottom-up and top-down) high-level approaches with various human-computer interaction (HCI) techniques were selected for taking the requirements and to design the system. The top-down approach is used to get information regarding the related works, while the bottom-up approach is used to get requirements from the practical work carried out in the field. We also used the experience knowledge of users and technical persons working in the emergency or disaster scenarios to get more user-centered focus. The work done according to the selected approach is as follow:

- **Bottom-up approach** A concrete case study of emergency management in the Calabria Region was conducted (cfr. Section 2.1). Potential users were

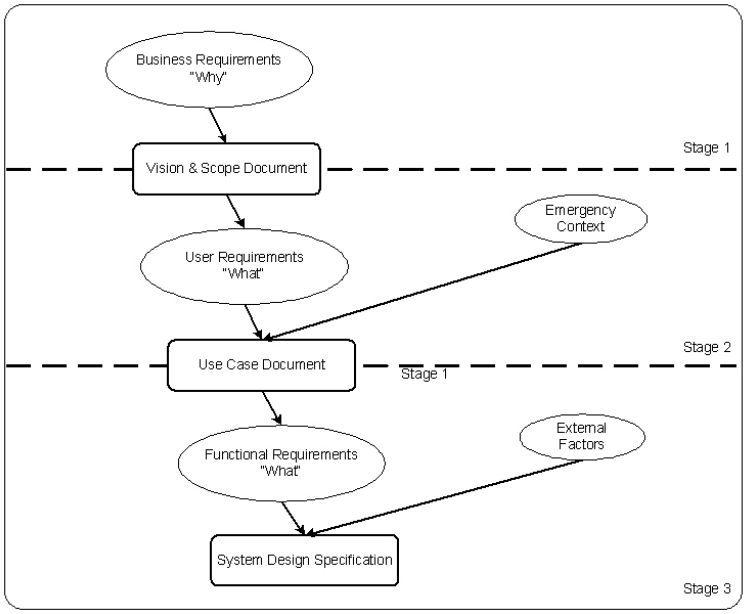


Figure 4.2: Stages within the requirements elicitation process

intensively involved in this project phase according to the international ISO standard 13407 [1].

- **Top-down approach** On the one hand, we investigated European legislation, recommendations and initiatives with respect to emergency management (cfr. Section 3.3), and on the other hand, related European research projects were examined regarding the requirements analysis methods adopted, the concrete outcomes and their validity for the WORKPAD project (cfr. Section 3.4).

The requirements elicitation process per se is an iterative and interactive process incorporating at least three stages with dedicated outcomes [60]. Figure 4.2 illustrates these three stages and was slightly adapted from [59] to represent the activities conducted in WORKPAD. Basically it's possible to individuate three types of requirements which are all subject to three individual documents:

- Business Requirements;
- User Requirements;
- Functional Requirements.

The *Business requirements* (first stage) denote the "why": Why is a project or a new development necessary? These are the driving forces for a project and are usually easy to collect because these are apparent. Business requirements represent the basis for the user requirements.

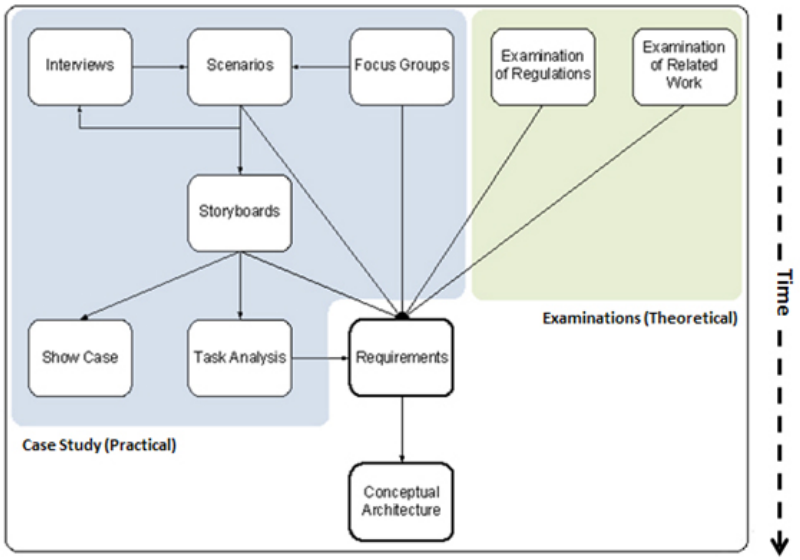


Figure 4.3: High-level overview of conducted activities for the the requirement elicitation process

User requirements (second stage) deal with one type of "what" information: What will the users finally be able to do with the system - such as tasks or goals they must be able to perform? In WORKPAD the specific characteristics of the emergency context on a broader basis beyond the knowledge of the users, which cannot be a complete knowledge, have to be taken into account by having a look at related activities, initiatives, and works. The user requirements are usually acquired by user scenarios, storyboards, or use case descriptions.

Taking this as the basis, *functional requirements* (third stage) are finally derived, which describe another "what": What are the developers of the system supposed to build? These are the traditional requirements that specify the functionality of the intended system on a more fine-grained basis. In this third stage, other external factors must be considered such as the available technology, realistic estimations, feasibility, or costs which have to be analyzed by additional studies or by empirical data of experienced stakeholders. The functional requirements represent the principal communication means between users, developers, designers, testers, and further project stakeholders, providing the common understanding about the intended system.

Figure 4.2 shows the dependencies and sequences of the various types of requirements and documents. The dashed horizontal lines denote the borders between the different states of the requirements elicitation process. By this methodology, requirements are steadily refined - through frequent iterations and user feedback - until the system is described thorough enough to establish the software specifications.

In the WORKPAD project, the concrete procedure of the conducted activities can be summarised as follows: after the definition of the potential users and user groups of the Calabria, two iterations of interviews and two focused user workshops were conducted, which resulted in a better understanding of the real end-users' problems and their needs (business requirements). Subsequently, with steady user feedbacks realistic scenarios were designed (earthquake and flood). Particular and concrete instances of the rather broadly defined scenarios are called storyboards and were derived from scenarios. The usage of storyboards in the WORKPAD project is threefold:

1. some of them are implemented in the final showcase,
2. they serve as a basis for the task analysis, and
3. specific requirements are directly derived from the storyboards.

The task analysis, in turn, also serves as an important input for the requirement collection process. From the Calabrian case study on the one hand and the theoretical work (examinations) on the other hand, generally applicable requirements for an emergency management system that shall basically be applicable in the whole of Europe were derived. The collected requirements suggested decisions for the design of the conceptual architecture (cfr. Section 2.2).

Figure 4.3 gives an overview of the performed activities, the dependencies and their temporal sequence (from top to bottom) until even devising a conceptual architecture of the WORKPAD project.

4.2.1 HCI Techniques Deployed in the Calabrian Case Study

As described in the previous sections, various human-computer interaction techniques have been deployed during the analysis of the case study of the Calabrian emergency system to capture relevant requirements for the WORKPAD project. Figure 4.4 gives a more detailed and technical overview of the methodology used only for the practical part of the requirements elicitation process of WORKPAD and depicts the several phases and their interrelations. The phases are comprised of the definition of user groups, development of scenarios, task analysis, requirements derivation, use case definition and finally analysis of the required WORKPAD system components.

In WORKPAD, the slightly adapted Scenario-based Requirements Analysis Method (SCRAM) [52] has been used in order to get a realistic understanding of the user's problem context, to derive early requirements that have served as a basis for further HCI techniques such as storyboards and hierarchical task analysis (HTA), to design the showcase, and later on to evaluate the WORKPAD approach. The SCRAM used in WORKPAD comprises four sub-phases:

1. Initial requirements capture and domain familiarization (i.e., business and early user requirements analysis) by interviewing, conducting focus groups and developing scenarios.

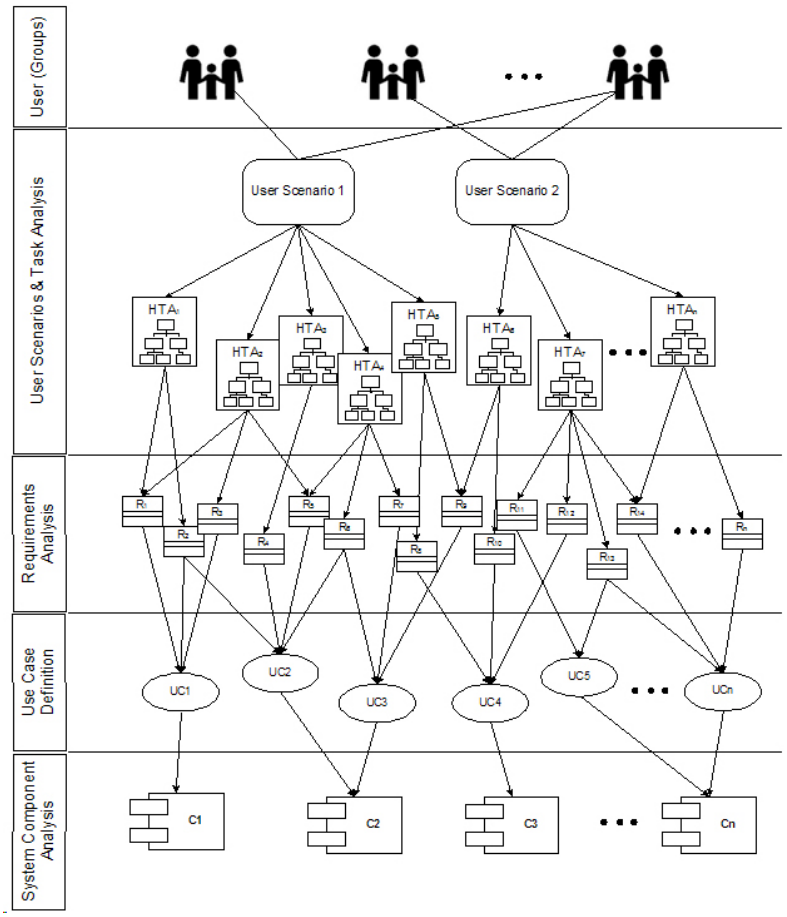


Figure 4.4: Methodology deployed in Calabrian case study

2. Design visioning by storyboards and HTAs to provide a more concrete impression of the future functions for users and system engineers by instantiating concrete facets of scenarios.

3. Requirements exploration (i.e., analysing feedback from stakeholders to current status of requirements by using scenarios, storyboards, paper-based or real early prototypes, or mock-ups).

4. Prototyping and requirements validation by more functional (horizontal or vertical) software prototypes representing a facet of the intended system to acknowledge requirements, respectively to agree upon necessary refinements or changes.

It starts with personal interviews of the potential users and possible workshops, which gives an opportunity for requirement engineers to better understand the tasks of the users. Interviewing is a guided conversation that involves structured or unstructured discussion between engineers and potential users of the system; this is

the most frequently used technique. This phase results in a clear definition of the user groups and in an overview of the current working situation, responsibilities and tasks of the potential users.

In the next phase, the engineers and potential users work close on the development of scenarios. Scenario building is an inexpensive and quick method for the collection of requirement and task information, and allows users to create a context for their requirements and tasks. An advantage of this method is that it does not provide any prioritization of requirements and tasks. We got scenarios through storytelling: users describe situations through stories. Stories are described in a "free-text style"; there is no formalization, e.g. structure of the processes, required in this phase. The most important result from the scenarios is a deeper understanding of the differences among several users' groups and their basic work flows performed within the organizations [22]. These scenarios serve as basis for the specification of functional requirements and task analysis.

Task analysis aims at showing an overall structure of the main user tasks; it includes the overall users' responsibilities in processes, goals to achieve and tasks that users intend to perform to achieve goals. An approach, known as HTA [3], divides high level tasks into their constituent subtasks which, in turn, are further subdivided up to a given level of details. HTA must be independent from the application, the planned system or other techniques used to perform the tasks. So, it is easy to allocate tasks into whichever application, and it enables to easily develop a conceptual model for them.

Scenarios and task analysis give the needed input for the user requirement analysis. User requirements allow to define :

1. problems each user meets performing her/his tasks,

2. solutions s/he has in mind to solve problems and

3. users' real needs (that is the functionalities that systems, both computer-based and not, have to provide).

In general, starting from users requirements, it is possible to distinguish between functional and non-functional system requirements. Functional requirements identify the characteristics and requirements posed on the target applications or systems, whereas non-functional requirements specify global constraints on how the software operates or how the functionalities are exhibited. Once functional requirements (what the system should do) are described in form of use cases, non-functional requirements (performance, reliability, efficiency, security, safety, etc.) can be added.

All the artefacts developed and the project outcomes obtained with the purpose to capture user and system requirements in the WORKPAD project are detailed in Chapter 5.

4.3 Evaluation and Validation Methodology

The objective of creating a validation and evaluation plan is to carry out an effective assessment of the project outcomes. This is necessary for evaluating and validating the project activities and corresponding results from several distinct perspectives such as individual technological components, as well as the entire system being developed in the project.

The WORKPAD evaluation and validation methodology refers to the User-Centered software development process [1], which means that users are actively involved throughout the whole software development process. This Section describes the overall evaluation and validation methodology adopted and the testing methods used in WORKPAD in order to ensure a user-driven development of the system.

The Section is structured as follows: at first it describes generally how usability evaluations have been performed in WORKPAD. Then the concrete evaluation and validation methodology is shown, providing information about the different types of user tests performed within the scope of the WORKPAD project. Finally, it discusses some details about the preparation of the WORKPAD showcases.

4.3.1 Overview of the User Tests Methodology

This subsection summarizes the methodology deployed in the WORKPAD project to involve users and to get their feedback in various forms, in order to carry out an effective assessment of the project outcomes. For the evaluation activities in WORKPAD we mainly used qualitative usability evaluation methods like feature inspection, observation of users while they perform different tasks, cooperative evaluation and questionnaires which tell us details about the user satisfaction. Concerning the number of test persons, Robert Virzi claimed that about 80% of the known usability problems could be discovered with 5 testers. 3 testers would find the most severe problems [58]. Also Nielsen and Landauer confirmed that 5 users discover approx. 80% of usability problems (see Figure 4.5). With 3 testers they showed that 70% of usability problems could be found [42].

Figure 4.5 shows with a curve the relation between the usability problems founded and the number of test users. In accordance to this, in WORKPAD we carried out several investigations in order to discover usability problems and give improvement recommendations to the developers.

The following types of user tests were accomplished:

- Online Pre-tests.
- Controlled experiments.
- Cooperative evaluation.

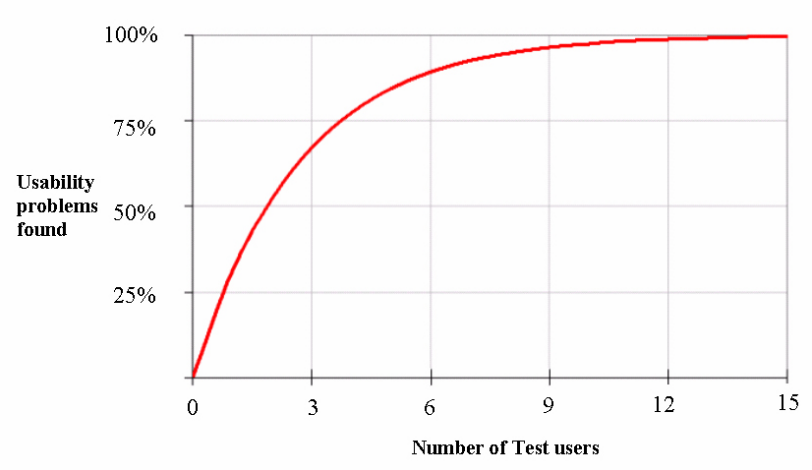


Figure 4.5: Detected Usability Problems vs. Number of Test Users

- Test with external users.
- Showcase without the WORKPAD system.
- Showcase with the WORKPAD system.

Figure 4.6 gives an overview of the various types of user tests executed in WORKPAD. The figure also shows the outcomes and how these tests are related to each other. The tests are chronologically ordered (from top to bottom) along the lifetime of WORKPAD.

The following subsections provide details about each of the deployed type of user tests; instead, a description of their results, conclusions and recommendations is presented in Chapter 6.

4.3.2 Types of Users Tests

In WORKPAD two usability tests with selected users from the Calabria Homeland Security Department were performed. They were executed prior to the final showcase of the project in order to ensure a user-driven development of the WORKPAD system. The results of the user tests have been evaluated using qualitative evaluation methods, and from their results improvement recommendations for the developers have been derived. The tests took place with a selected number of users during the design phase of the software development process in order to get feedback from the users with regard to the following points:

- Test whether the requirements were understood correctly.
- Test whether the main functionalities met the users' demands.
- Test whether the mock-up or prototype was easy understandable and usable.

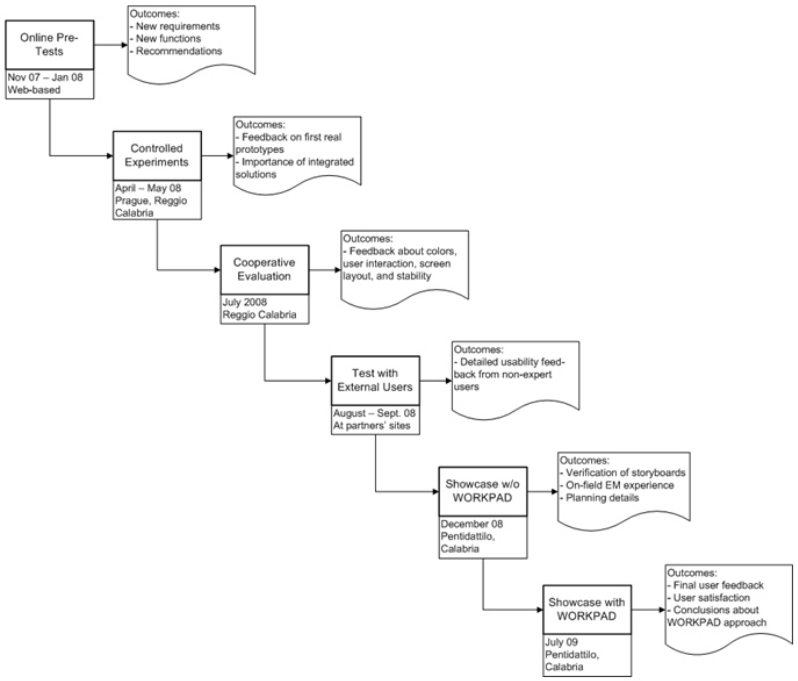


Figure 4.6: User Tests Methodology in WORKPAD

- Test whether specified tasks could be performed easily.

In order to perform the user tests we used the following methods: qualitative online questionnaire and cooperative evaluation accompanied by a semi-structured interview including a questionnaire.

Usability test with online mock-ups

The first usability test took place when the mock-ups of the different components were ready. The test including the mock-ups was performed with an online questionnaire.

Usability testing with prototypes using cooperative evaluation method

The second usability test was performed with stand-alone component-based prototypes. The qualitative test was executed within the scope of the project meeting in Calabria with four selected users. The following evaluation procedure was applied:

- *Cooperative evaluation test with users:* The prepared tasks had to be accomplished by the users using different WORKPAD components. The cooperative evaluation was performed with four users and lasts about four hours on the whole. The evaluator guided the users through the test and performed

the evaluation in a cooperative manner meaning highly interactive. The user test was audio and videotaped.

- *Semi-structured interview with questionnaire:* After the user finished the tasks (s)he was asked to watch the video and to comment it. The evaluator took notes of the user's comments. After looking through the video a semi-structured interview which includes a background questionnaire was performed with the users.

The user test was analysed and served again as recommendation input for the system developers.

Controlled Experiments

Whereas usability testing observes a single system or one design, controlled experiments test and compare user interfaces under controlled experimental conditions. In WORKPAD, we performed one controlled experiment prior to the showcase with the WORKPAD system in Calabria. This was to ensure a proper working and interfacing of the different WORKPAD components.

WORKPAD Showcases

With the term "WORKPAD Showcase" we intend the implementation of a concrete emergency management scenario in a real world context. This scenario was performed in a realistic setting at a simulated emergency site with emergency operators. Particular storyboards and selected tasks out of the emergency scenario have been showcased. In order to evaluate the system at the best, we have performed two identical drills (each one with different teams, but identical user profiles of the participating users).

In the first drill operators acted with their current knowledge and technologies (i.e., without the support of the WORKPAD system). This first showcase was a proof of concept of the design of the various storyboards for the test with the system. It was very helpful to watch and document the work of the different emergency organizations and to get a better idea for the second showcase with the WORKPAD system.

In the second drill, organized some months after the first one, the overall WORKPAD system is evaluated by selected users. Before to start the final showcase we set up the whole WORKPAD system and identified whether all system features function with regard to the drill. Only users who had the same user profile as the ones in the drill without WORKPAD participated in this second drill. Each user, after have obtained instructions about the use of the system, received a personal digital assistant (PDA) with the WORKPAD system pre-configured and then performed several tasks.

The users' test activities were videotaped so that a detailed analysis could be accomplished after the test. The test results gave also input for the formulation of

recommendations for the developers.

Chapter 5

Requirements Artefacts

In the WORKPAD project, twofold (bottom-up and top-down) high-level approaches with various human-computer interaction (HCI) techniques were selected for taking the requirements and to design the system. The top-down approach has been used to get information regarding the related works. In this sense, first we have investigated European legislation, recommendations and initiatives with respect to emergency management (cfr. Section 3.3), and on the other hand, we have examined related European research projects regarding the requirements analysis methods adopted, the concrete outcomes and their validity for the WORKPAD project (cfr. Section 3.4). Instead, the bottom-up approach has been used to get requirements from the practical work carried out in the field.

This Chapter aims to describe the design and the implementation of all the artefacts necessaries to conduct the mentioned bottom-up approach, which consist in a concrete case study of emergency management held thanks to the help of the operators of the Calabrian Civil Protection [49].

The Chapter is organized as follow :

- Section 5.1 identifies the potential users of the system, selected among the Calabrian organizations involved in Emergency Management.
- Section 5.2 shows the design of a semi-structured interview, used to have a better understanding of the real end-users' problems and needs.
- Section 5.3 describes the design of two realistic scenarios (earthquake and flood).
- Section 5.4 presents a Task Analysis conducted on the basis of ten storyboards derived by the scenarios.
- Section 5.5 details the final list of User Requirements of the project.
- Section 5.6 furnishes a use-case oriented analysis of requirements.

- Section 5.7 details the list of System Requirements stemmed by the analysis of User Requirements and by the knowledge of the current technology.

5.1 Defining Users and Categorization of User Groups

The first task conducted during the case study of Calabria was to identify the potential users and user groups. In order to understand how Civil Protection works during emergency management, we interviewed some officers and actors which are really involved during emergencies. In collaboration with the Civil Protection Department of Calabria, there was identified two typologies of users: Back-End and Front-End users.

- ***Front-End users:*** They are all the operators acting directly on the field during emergencies/disasters (ranging from firemen to voluntary associations).
- ***Back-End users:*** They are all the operators who manage the situation from control rooms, by providing goals/instructions/information to Front-End operators.

Additionally, the same users can be classified orthogonally with respect to the various organizations potentially involved in emergency situations, such as police, fire brigade, medical board, army, etc. Furthermore, every organization comprises different roles, commander, team leader, team member, radio operator, etc.

A exhaustive picture about the organizations and users involved during an emergency in Italy is detailed in the following subsection 5.1.1.

5.1.1 Involved Actors, Actions, Tasks and Duties

Managing an emergency means handling some actions and procedures which can be summarized as:

- evaluation of event's effects;
- emergency run-up;
- back to normality.

Evaluation of effects is fundamental to know exactly all problems to be solved and to plan resources and actions to be undertaken. The urgent first aids must aim at contrasting the following negative effects:

- negative effects on population (victims, wounded people and possible panic situation):
  - aid to population for safeguarding people's safety;
  - first hospitalization, attendance e victualling of population involved;

  - building emergency edifices such as tents, roulottes, containers or whatever.

- negative effects on structures (building unfit for use or collapse):
  - complete check-out of dangerous or collapsing buildings;
  - urgent procedures of building consolidation or evacuation;
  - damaged building survey and habitability test;
- negative effects on essential services:
  - carry on technical services in order to face emergency, to limit damages and to keep environment, goods and life integrity.
- negative effects deriving from the persistent danger:
  - discarded material recover;
  - economic evaluation and reconstruction suggestions;
  - restore normal activities.

Normality restoring begins with dwelling houses and social-economic environment reconstruction/consolidation; it stops after a short or longer period, when all pre-earthquake conditions are restored.

In the following, we report the duties of the various actors involved in the Emergency Management in Italy.

Prefect's Office (UTG)

- summon UTG Civil Protection office staff;
- summon CCS members and activate the operative hall for the supporting functions (according to the "Augustus method");
- make a first emergency evaluation;
- set up the necessary COMs and COCs;
- establish alternative emergency radio links between CCS and COMs and between COMs and COCs;
- inform various political actors, including the National DPC, the Ministry of the Interior, the Reggio Calabria Province Presidency;
- in relation to the emergency situation, UTG will ask:
  - to the Ministry of Interior, the employment of Police and Carabineers divisions equipped for public aid;

  - to the Calabria Regional Military headquartes, the cooperation of Army Unities;
  - to the Fire Brigades and the Ministry of Interior, the activation of Assistance First Aid Centres (CAPI) for the requested emergency needs.

In general, the UTG will ask interventions falling under the respective competences, relative to the best use and disposition of men, means, services, materials, setting up camps, tents, provisions, shelter for animals, employment of helicopters and aeroplanes for photos and survey and what is necessary to satisfy the population needs in an emergency. Through mass media, UTG will give population detailed information about the events with some behaviour rules to be adopted.

Mayor and Civil Protection Operative Unit Responsible (C-PC)

- set up COC and the support functions of the Augustus method;
- start a first aid through local means and aid squads;
- give immediate effectiveness to Civil Protection municipal plan;
- set up alarm systems (loudspeakers, bells, sirens, etc);
- move population into safe areas, in cooperation with CCS and COMs;
- refer to CCS about identified, unidentified corpses and missing persons;
- find beds in public structures and temporary buildings for the homeless;
- refer to UTG about the needs of camp tents, roulottes and other sheltered accommodations;
- provide people with food, water and staple commodities in cooperation with COMs and CCS; eventually using emergency measures with public concern;
- inform COMs about possible blackout and telephone problems;
- relate CCS and COMs about the state of municipal and nearby roads;
- establish contacts with CCS or COMs sanity referents for the organization of sanitary and veterinary fittings;
- coordinate the getting ready of public offices and essential public services;
- care about the safety of documents and files of municipal and public offices;
- care about the system of the emergency service signs for the best help to population;
- inform COMs about damages to cultural and artistic patrimony, or public works, or private goods;

- coordinate the storage of food, materials, clothes from public donation and its rational distribution among population.

Essential Services

They include the infrastructure operators providing services such as the distribution of energy, gas and water. They also include transport, telecommunication service providers, etc. Essential Services are characterized by holding widespread diffusion all over the territory.

- **National Roads Administration (ANAS)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - identify, together with Police, Traffic Police and Province Police, alternative inter-municipal routes;
  - provide proper sign road system;
  - check the stability of handworks;
  - care about the practicability of state roads with the provincial and municipal one;
  - ask for specialized road firms when necessary;
  - activate, when necessary, radio set at U.T.G.
- **Electrical Energy National Corporation (ENEL)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - mobilize the technic staff for a h/24h operative hall service;
  - send survey squads linked with the operative hall;
  - ask for other technical squads from other provinces if necessary;
  - ask for specialized firms when necessary;
  - set up, on U.T.G. request, temporary flying electric lines where necessary;
  - check the security of all the electric installations.
- **Italian Railway System (RFI)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - check and re-establish the damaged railway system;
  - authorize, when asked, a railway line link with a Telephon Operator;

  - give, for all the necessities, railway carriages and vagons to be parked in station dead-end tracks;
  - send covers, medicines, sanity materials stored;
  - see to water supply through tanks;
  - cooperate at the best with the civil and military authorities to send the hit zones men, means and materials;
  - give the right way to hospital trains, aid carriages, and Civil Protection units.

- **Airports (e.g., SOGAS for Reggio Calabria)**
  - verify the integrity of airport structures for safe flights;
  - assure a complete answer to the airport activities h/24;
  - make a census of all the aircrafts available and ready to take off ;
  - establish contacts with UTG operative hall to assure about aircrafts and refuelling;

- **Telephone Operators (TLC)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - check the telephone networks and restore in favour of the bodies involved in the emergency;
  - set up new telephone lines according to U.T.G. operative hall needs;
  - create a link with railway telephone lines;
  - set up new public telephone boxes;
  - ask for other technical squads from other provinces, if necessary;

- **Gas (SNAM)**
  - mobilize the staff and the means available;
  - dispose for a check and re-establishment of gas system;
  - stop the supply of gas to buildings damaged or evacuated.

- **Aqueducts (e.g., SoRiCal in The region of Calabria)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - dispose for a check and re-establishment of water system;
  - stop the supply of water to buildings damaged or evacuated.

- **Italian Dams Register (RID)**;

  - check the territory storage capacity, and report every abnormality to UTG, the Fire Brigades (VVFF), the Mayors and Carabineers;
  - in agreement with the Prefect and after a careful analysis of the storage capacity situation, take initiatives and measures to safeguard the public and private safety.

- **Traffic and Transport Control Provincial Office (MCTC)**
  - gather in the deposit all the means of transport available and inform UTG about their total number and type - all of them must be ready to run;
  - recall all the drivers, even at rest, to assure the complete readiness of the means of transport gathered;
  - inform all the time CCS about the number and the type of motor vehicles available.

- **Post and Telecommunication (PI)**
  - rapid re-establishment of telegraphic and postal services;
  - set up of new temporary postal offices when necessary.

Public Security Organizations

Public Security Organizations arrive as the first organizations on the place of the emergency. Usually citizens signal to them dangerous situations. They typically go immediately to the place to analyze the situation and, if it is not a false alarm, inform the prefecture. During emergency, each public security organization approaches the situation from its own specific viewpoint. The Police and Carabineers guarantee the maintenance of public security and the laws. The Urban Police deal with traffic. A specific mention is needed for the Fire Brigade: according to the law, they are in charge to coordinate actions on the field and suggest to the prefect what to do. Every organization may have a control room which communicates on one side with both CCS and COMs, and on the other side with the operators on the field. Public Security Organizations are:

- the **Fire Brigade (VF)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - send to the disaster zone service squads and the necessary means;
  - ask the Fire Brigade Regional Department for the Regional Mobile Column and/or forces from other Regions or headquarters;
  - ask the Ministry of Interior competent Department for helicopters for emergency transfer and survey;

  - inform all the time UTG operative hall about the situation;
  - maintain contact with Fire Brigade Regional Department;
  - give directions about traffic discipline and showing alternative routes to be used by the public first aid means of transport;
  - activate radio sets at UTG when necessary;
  - Fire brigades make first interventions; in particular local firemen, then regional firemen, then regional flying squads move on territory and install base camps; they supply first aids, than first inspections to remove and safeguard dangerous buildings; and answer to direct call for performing local on-the-spot investigations.

- the **Carabineers [Carabinieri - CC]**
  - send representants to UTG and COMs, COCs for its own competencies;
  - activate radio network links;
  - inform all the time UTG;
  - check through local Carabineer Stations eventual Mayors (COCs) problems.

- the **Revenue Guard Corp (GDF)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - set up services for public and patrimony law and order;
  - provide the prevention of looting;
  - activate radio sets at UTG when necessary.

- the **State Forest Corp (CFS)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - cooperate with the Mayor (COC), with the competent regional bodies and the Consortium of Communes in mountain areas (CMs) as for an alternative service of distribution of drinkable water;
  - promote activities of safety check in the areas of disaster;
  - activate radio sets at UTG when necessary.

- the **Police (POL)**
  - send representants to UTG and COMs, COCs for its own competencies;

  - carry out surveys of the hit areas (possibly with helicopters, if available), on Prefect's request, and inform about;
  - set up services for public and patrimony, maintenance of law and order;
  - prevent measures against looting;
  - recover and custody goods and valuables and their restitution to owners;
  - activate police radio sets, at UTG when necessary.

- the **Coast Guard Corp (CP)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - make a check of land and sea areas as due, with particular attention to harbour structures, together with other institutional forces endowed with watercrafts;
  - make a census of the damages produced to ships, boats, sport crafts present in the hit harbour;
  - activate radio sets at UTG when necessary.

- the **Army (EI)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - establish actions and men at different operative levels;
  - activate the Emergency Intervention Plan made by the competent military bodies on Prefect's request together with Army Units:
  - send first aid forces (men and means for the saving of human lives);
  - evacuate the wounded people;
  - evacuate population and goods;
  - set up camp hospitals, camp tents and roulottes;
  - supply catering for people;
  - provide water supply;
  - provide food, medicines, clothes;
  - manage, collection and distribution of materials;
  - activate radio sets at UTG when necessary.

Health Board

The *Local Health Unit* [*Azienda Sanitaria Locale - ASL*] has the role assure, in its own territory, the levels of assistance established by the regional sanitary plan. Among its duties, there is the prevention of illnesses causes, the reduction of risks and accidents in the work and life environment. Its structures that are involved in emergency management are:

- the **Sanitary emergency service (118)**
  - send representants to UTG and COMs, COCs for its own competencies;
  - mobilize staff according to an internal emergency plan;
  - inform about all the beds available;
  - assure the helicopter aid together with other institutional forces endowed with air means;
  - supervise:
    - first aid services;
    - water drinkableness;
    - burying of corpses;
    - hygienic-sanitary safety;
    - vaccinations;
    - transfer of wounded people to hospitals;
    - check of blood supplies;
    - disinfection and disinfestor of the countries;
    - burying of carcasses;
    - catching of wandering dogs;
    - check of dog population in general.
  - provisions of meals and its transfer to the hit zones;
  - sanity assistance to the zootechnic patrimony;
  - eventual vaccine prophylaxis and supply of sanity medicines.
- the **Red Cross**
  - send representants to UTG and COMs, COCs for its own competencies;
  - help the transfer of wounded, elderly disabled people, and children;
  - set up and manage first aid socio-sanitary assistance;
  - provide a census of wounded and dead people;
  - help the research of missing people and their joining again with families;

  - help the management of camp hospitals;
  - help the setting up of camp tents and roulettes;
  - help the storage and distribution of food, medicines, clothes, belongings.

- and the various **hospitals (H)** present in the territory of its own competency.

Voluntary Service

Many Voluntary associations cooperate with the Civil Protection in their needs for first-aid response. The voluntary organizations include qualified social and work groups of modern society; this is a very essential aspect mainly in great emergencies: there intervention success depends on the contribution of many different specializations (from doctors to engineers, from nurses to electricians, from chefs to carpenters). Some organizations are characterized by high specializations, such as dog unit groups and free divers, as well as radio amateurs, speleologists, voluntary people for wood anti-fire, etc. The Voluntary associations cooperate in removing ruins, searching missing people, coordinating a correct transfer of population, setting up tents, distribution of food and what necessary to face at the best the emergency, in agreement with the institutional forces. They assure the presence and the intervention in the zones of disaster, only if they are considered self-sufficient (i.e., enough skilled/trained) by Civil Protection staff.

5.2 Conduction of Initial Interviews

In order to learn users, with their problems and expectations, we developed semi-structured interviews based on the user group definitions. The interview guideline consisted of a general introduction part, the instructions for the moderator and a description of the WORKPAD project; and the second part was represented by the set of questions, divided into three sub-parts:

1. basic data about the interviewee;
2. more general questions and
3. specific questions depending on the user group class that resulted in two variants of questionnaires (one version for the potential Front-End users and a second version for the Back-End users).

We used open-ended questions, allowing each interviewee to answer to any question as he/she prefers. Finally, the last part of the questionnaire was a discussion, where the interviewee could reflect again his answers and the moderator had the possibility to ask further questions for clarification. We conducted 32 interviews, approximately 45 min length of each, from the officers and generic actors of

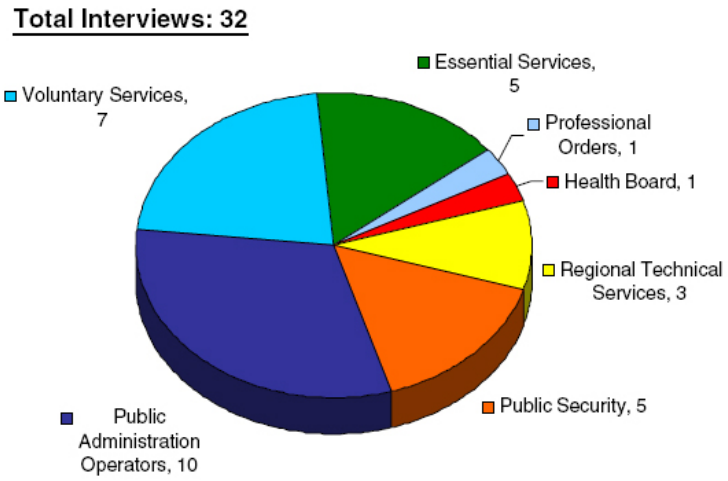


Figure 5.1: Number of interviews in each category

the most important organizations (e.g. Harbour Office - Coast Guard, Police Headquarter, Civil Protection, State Forest Corp Provincial Headquarters, Italian Red Cross of Calabria region, etc.) involved in emergency management in the Calabria region. We divided these organizations into seven different categories.

Figure 5.1 shows a pie chart in which each slice represents the number of interviews taken in each category. During each interview, the moderator, as the leader of the interview team, was responsible to tell to the interviewee the purpose of the interview and the project and to ask the questions according to the guidelines of the interview template. The answers were recorded by another member of the interview team, who was also responsible to make detailed notes into the specific form. The interviews situations were video-taped by a third (technical) member of the team so that all statements were backed up. The main purpose of the interviews was to identify the users and their activities to manage disasters for which we asked them to imagine a realistic situation that could happen during their work and to describe the tasks to face it.

The description presented here was how the first application of this HCI technique was applied. The following subsection provides the complete questionnaire template that was used while conducting interviews.

5.2.1 Interview Guidelines and Questionnaire

Instructions for the Moderator

```
The Interview Guidelines serve as a basis for the
moderator in order to get detailed information on the
user requirements.  The moderator is the leader of
the interview and guides the potential user through
```

the personal interview by asking questions that are specified in this document. The answers are recorded by another person who also takes part in the interview and makes detailed notes into the specified form. The interview situation is videotaped and tape recorded by a third (technical) person so that all statements are backed up.

Participants

Moderator, potential user, technical person responsible for tape recording and video taping, support person taking interview notes.

Duration

Approximately 45 minutes

Background: the WORKPAD Project

The moderator gives a short summary on the WORKPAD project to the users. The WORKPAD project aims at building and developing an innovative software infrastructure (software, models, services, etc.) for supporting collaborative work of human operators in emergency/disaster scenarios. In such scenarios, different teams, belonging to different organizations, need to collaborate to reach a common goal; each team member is equipped with hand-held devices (PDAs) and communication technologies, and should carry on specific tasks. In such a way we can consider the whole team as carrying on a process (macro-process), and the different teams (of the different organizations) collaborate through the "interleaving" of all the different processes. Each team is supported by some Back-End centre, and the different centres need to cooperate at an inter-organizational level to reach an effective coordination among teams. Each team is referred to one specific organization, which is the only agent in charge of controlling its teams, and the teams thus form a virtual organization. The project will investigate a 2-level framework for such scenarios: a Back-End peer-to-peer community, providing advanced

services requiring high computational power, data and knowledge and content integration, and a set of Front-End peer-to-peer communities, that provide services to human workers, mainly by adaptively enacting processes on mobile ad-hoc networks.

Basic Data

- Date :
- Name of the interviewed person :
- Organisation :
- Position in the organisation :
- Moderator :
- Present persons :

Questionnaire

The moderator asks potential users the following questions:

Question 1 :

What are your main responsibilities within this organisation?

Question 2 :

In what kind of emergencies is your organisation involved?

Question 3 :

What is your role during an emergency? In which phase of an emergency are you involved?

Question 4 :

Do you know the statistical frequency according to which an emergency happens in your territory?

At this point (it depends on the user), the interview is divided in two trunks: the first concerns

Front-End users, and the second concerns Back-End users. The main idea is to immerse the user in the context of an emergency. It means that we have to investigate the steps that the user performs when preparing himself to face the emergency (when he/she gets a call related to an emergency), until the moment in which he/she has to act really. In this way, we create an "implicit scenario" for the user (he/she believes to be in an emergency situation), and he can answer in the way he wants to.

Front-End User

Shortly after the emergency has happened

Question 5a :

Which steps do you perform shortly after the emergency has happened?

Question 6a :

What kind of information (related to the emergency) do you get from the control centre?

Question 7a :

How long is the Front-End team actively involved in this phase of the emergency (average) ?

Question 8a :

What kind of information do you exchange with other members of the team during the transport to the place where the emergency has happened?

During the emergency

Question 9a :

Describe the composition of the team and the various roles of the team members allocated to them during the emergency.

Question 10a :

What kind of technical devices do you currently use in emergencies?

Question 11a :

How do you communicate with the other team members and the Back-End centre?

- Does your team use a separate communication channel?

Question 12a :

What kind of technology do you currently use in/after emergency situations?

Question 13a :

What kind of information (and in which form) do you exchange with the team leader?

Question 14a :

What kind of information (and in which form) do you exchange with the Back-End centre?

Question 15a :

Do you co-operate with members of other organizations ? (for example police, etc.)?

- Do you exchange information and/or data?
- Do you share a common technology?

Back-End User

Shortly after the emergency has happened

Question 5b :

Which steps do you perform shortly after the emergency has happened?

Question 6b :

How much time are the Back-End team actively involved in this phase of the emergency (average)?

Question 7b :

What kind of information do you send to Front-End operators, who have to prepare them to face the emergency?

Question 8b :

In what way do you obtain such information and in which format?

Question 9b :

Please, describe the structure of your organisation and the various roles assigned to the team members in this phase of the emergency?

During the emergency

Question 10b :

What kind of technical devices do you use for the communication with the Front-End operators?

Question 11b :

What kind of communication technology do you use?

- Does your organisation use a separate communication channel?

Question 12b :

Does the communication take place with a particular team member(s) or can you communicate arbitrarily with everybody (how strict are the hierarchical and the communication structures defined within your organisation)?

Question 13b :

What kind of information do you send to the Front-End users?

Question 14b :

What kind of information do you receive from the Front-End users?

Question 15b :

Do you share technology and data with other organizations?

- Which kind of data / technology?
- In which way does this exchange of information take place?

The last questions are the same for every user

Question 16 :

Do you currently use Geographic Information Systems (GIS)?

- If yes, which software and data do you use?

Question 17 :

Do you think that the devices and technologies used to face the emergency are conform to the purpose for which they are used?

Question 18 :

What do you think would be a big improvement concerning the technology part?

- What kind of improvement would you propose?

5.3 Scenario Development and Targeted Interviews

The scenario-based requirements analysis method is a good way to develop a common understanding of the context, the activities and the problems that an organization has to face. The scenarios help us to think about the design in detail and notice potential problems before they happen. In the WORKPAD project, we concentrated on two scenarios: earthquake and flood, due to the fact that these are the most relevant ones in our context of study. We designed activity diagrams for both scenarios to describe how the end users would follow in order to face the emergency situation. Each scenario was developed to focus on a different phase of emergency: earthquake scenario covers the response phase, while the flood scenario covers the short-term recovery phase. Despite this difference, at high level the resulting flows of activities are the same in both of them. Scenarios have been organized as follow :

- giving a brief introduction to the scope;
- containing scenario title, relevant emergency phase, main goal, duration, actors, initial state, final state, and dependencies;
- designing UML activity diagrams depicting the sequence of involved high-level activities.

Moreover, due to the necessity of more detailed data, we conducted 14 further user interviews (not detailed here for reasons of space) to refine the proposed storyboards, which at that stage were not perfectly appropriate and detailed enough. These interviews were much more targeted: seven for the earthquake and seven for the flood scenario with specific questions. The result of these interviews served as input for the establishment of ten, more realistic storyboards (five for each scenario) and the subsequent task analysis.

The next two subsections shows the two scenarios used.

5.3.1 Description of the Earthquake Scenario

This sub-section is intended to study the necessary flow of activities that some organizations perform to manage the emergency situations concerning an earthquake. Figure 5.2 shows the description of all relevant aspects to be taken into consideration during the evaluation of the scenario. Through the Earthquake Scenario is treated the Response Phase.

The main focus of the Response phase during an emergency is to save people's life. Such a phase comes directly after a disaster happening and it is started by the National Department of Civil Protection (DPC for short). Whenever DPC recognizes in whichever form that a disaster breaks out, it tries to retrieve as much data as possible. This data can be retrieved in different way, depending even on the kind

Scenario	Earthquake
Phase	Response Phase
Main goal	First aid to population
Duration	2-3 days
Actors	National and regional Civil Protection Departments, Police (State Police, Carabinieri, etc.), Hygenic Public Health department, Voluntary Services, Transportation & Infrastructure (e.g., Railway) Providers, Fire Brigades, State Forest Corp
Initial State	Emergency incident → notification about seismic activities
Final State	Teams are present in the field and received appropriate commands
Dependencies	Predecessor to second phase (Short-term Recovery Phase)
Task overview	See Figure 5.3: Macro Description of the Response phase process

Figure 5.2: Description of the Earthquake Scenario

of emergency: by analyzing calls from common citizens, by specific sensor networks or by calling external entities, such as National Institute for Geophysics and Volcanology [Istituto Nazionale di Geofisica e Vulcanologia] - INGV in case of earthquakes. Starting from this data, DPC classifies emergency: national, regional or communal (level A, B or C). If emergency is classified as national, DPC handles it by itself. Otherwise, National DPC alerts regional DPC (RDPC) and delegates the control to it. RDPC arranges its regional unified operative control hall (SOUR - Sala Operativa Unificata Regionale).

By SOUR, RDPC can check anytime the overall situation. For each involved province, DPC opens a provincial control hall; in the meanwhile, Prefects establish one CCS per province. CCS is continuously headed and coordinated by Prefects or their delegates. CCS takes physically place in a special room, set up in "peace" time (i.e., before emergencies in a peaceful situation). During emergencies, the "political" chiefs of the involved organizations sit down in such a special room. Prefects delineate which zones are actually involved and, so, which COMs have to be opened. Then, prefects send delegates to head opening COMs.

In such an first phase, the COMs' organizations coordinate and send proper teams to affect areas to save people. Meanwhile, other teams are sent to establish first-aid infrastructures (such as field hospitals, tent cities) water and food in order to sustain affected communities. Teams saving people send back to their operative chiefs in COMs some information about people saved and still to be saved in order to determine whether other teams are necessary to progress in this phase. This

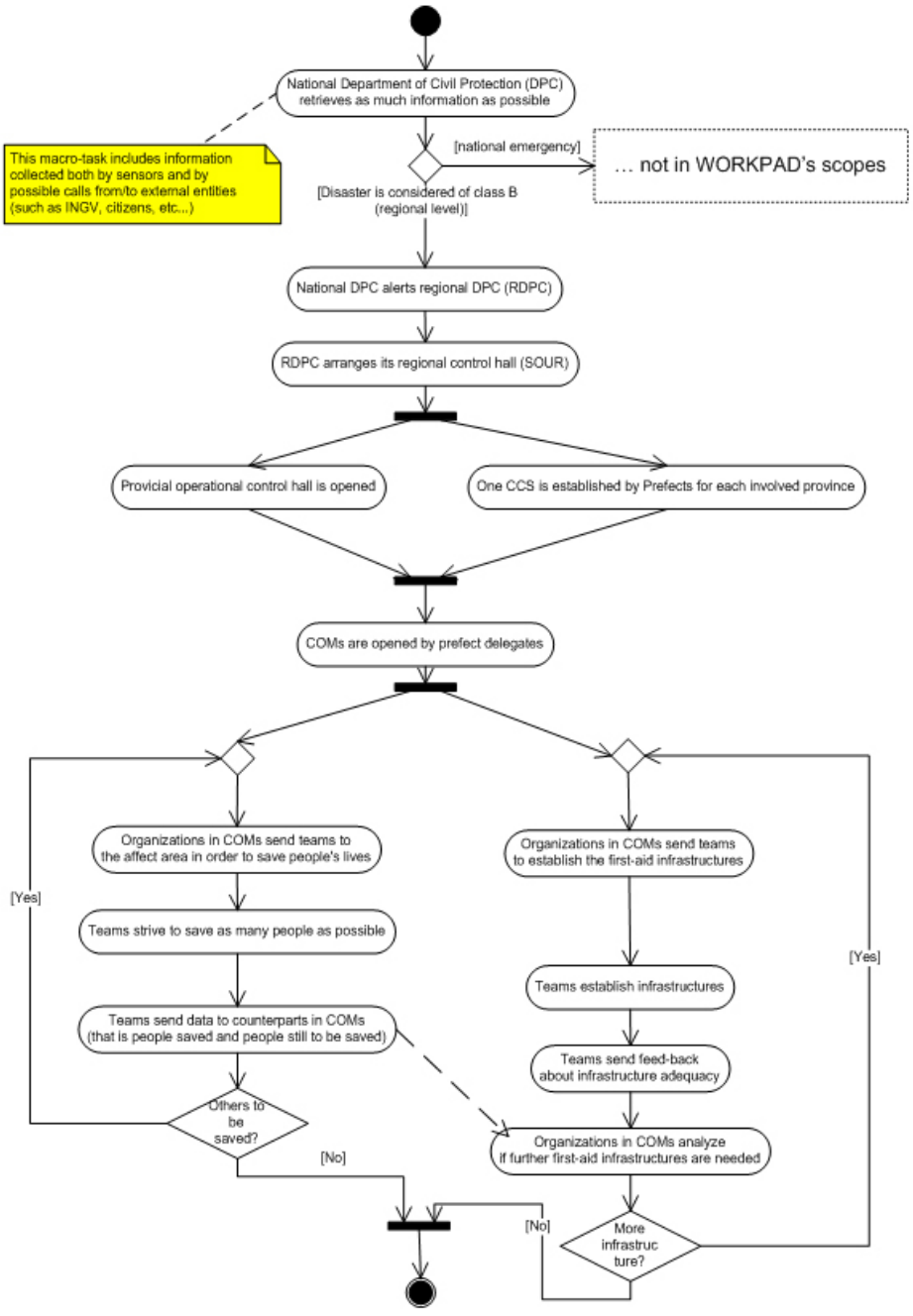


Figure 5.3: Activity Diagram describing the Earthquake Scenario

feed-back is also used by the COMs' organizations to learn even whether prepared infrastructure is enough, considering the amount of people in trouble, or, otherwise, other infrastructure is needed. Of course, this choice takes also into account (and perhaps above all) the direct information from teams establishing first-aid infrastructure. Figure 5.3 shows how this two branches (both the one about people rescue and the other about infrastructure establishing) are somehow cyclical. That is to underline rescue operations and infrastructure establishing go on until everybody is safely assured (or considered dead). First phase can last 3-4 days, as after it may be assumed that all people are safe or died.

5.3.2 Description of the Flood Scenario

This sub-section is intended to study the necessary flow of activities that some organizations perform to manage the emergency situations concerning a flood. Figure 5.4 shows the description of all relevant aspects to be taken into consideration during the evaluation of the scenario. Through the flood scenario is treated the Short-Term Recovery Phase.

Scenario	Flood
Phase	Short-term Recovery Phase
Main goal	Recovery of the affected area, restoring infrastructure/essential service
Duration	14 days
Actors	DPC (national and regional), Fire Brigades, Army, Police (State Police and Carabinieri), Hygenic Public Health department, Voluntary Services, Transportation & Infrastructure Providers, State Forest Corp
Initial State	Initial assistance is provided, situation is stabilised → living conditions can not yet be sufficiently provided
Final State	Basic living conditions can be provided up to a certain degree → CCS are closed
Dependencies	Ancestor to response phase and predecessor to further long-term recovery phases
Task overview	See Figure 5.5: Macro description of the Short-Term Recovery phase process

Figure 5.4: Description of the Flood Scenario

The Short-Term Recovery phase is intended to analyze the result of disaster happenings in order to schedule in following phases rebuilding activities and to make everything safe. The figure does not contain any information about CCS and COM settlement, since it's assumed this phase to come always after the first one.

This activity diagram is built from the viewpoint of exactly one organization: so, whenever it's not differently specified, macro-tasks are assumed to be executed by the same organization. The COMs' organizations determine jointly which are the buildings (like churches, schools, etc.) and other places to be checked and made safer. So, some organizations get assigned spots which information is needed about and they arrange teams which are sent. Teams send feedbacks to counterparts in COMs. Organizations analyze collected data and derive possible goals to be pursued (typical to make some place safer). During a joint meeting in COM (typically there are two meeting per day), all organizations propose their goals (possibly some goals contrast: in those cases, the prefect delegate settles questions). All goals are discussed together to uniform them. The result is that goals may change or may be assigned to other organizations with respect to those initially making the "suggestions". This can be caused by a sort of load balancing: it is possible some organizations have a lot of goal to be pursued, whereas others do not get anything to do. It is possible, as well, for an organization to get assigned to continue the work-flow in progress or to repeat it again. For each assigned goal, the organization arranges a team, creates a workflow to get the goal and sends the team to the proper place in the affected area to carry out the workflow. Every team, after carrying out the workflow, sends feedbacks to COMs (success of failure in getting the goal, other collected data, etc.). Possibly some goals are not completely clear and cannot be assigned to any time, since collected data is not enough. In this cases, in the meanwhile, other teams are sent by organizations to the area in order to collect other information. This data will be used in the next cycle to derive other goals to be pursued. The overview is shown in Figure 5.5.

5.4 Storyboards and Hierarchical Task Analysis (HTA)

The actual work, which the organizations perform, depends strongly on the disaster characteristics. In order to go deeply into "the mind" of rescue operators, we asked them to illustrate their own personal experiences in past-occurred disasters. These are called storyboards, and describe a specific emergency situation to be faced, taking into consideration some relevant conditions and furnishing a goal to reach. The storyboards are more concrete interpretations of a specific sub-part of a scenario and hence constitute a further refinement. The storyboards were presented in a structured way comprising the following information: actor(s), relevant emergency phase, initial state, relevant pre-conditions, final state, main goal, duration, and dependencies. Using these storyboards, we conducted a task break-down analysis through classical HTA technique [3]. The task analysis observes user behavior and focuses only on the strategy as a sequence of steps in order to reach the goal. So, it observes user behaviors: what they do to perform tasks without considering the reasons (the "*why*"). We stopped the HTA at the lowest level that allows a structured plan to be executed.

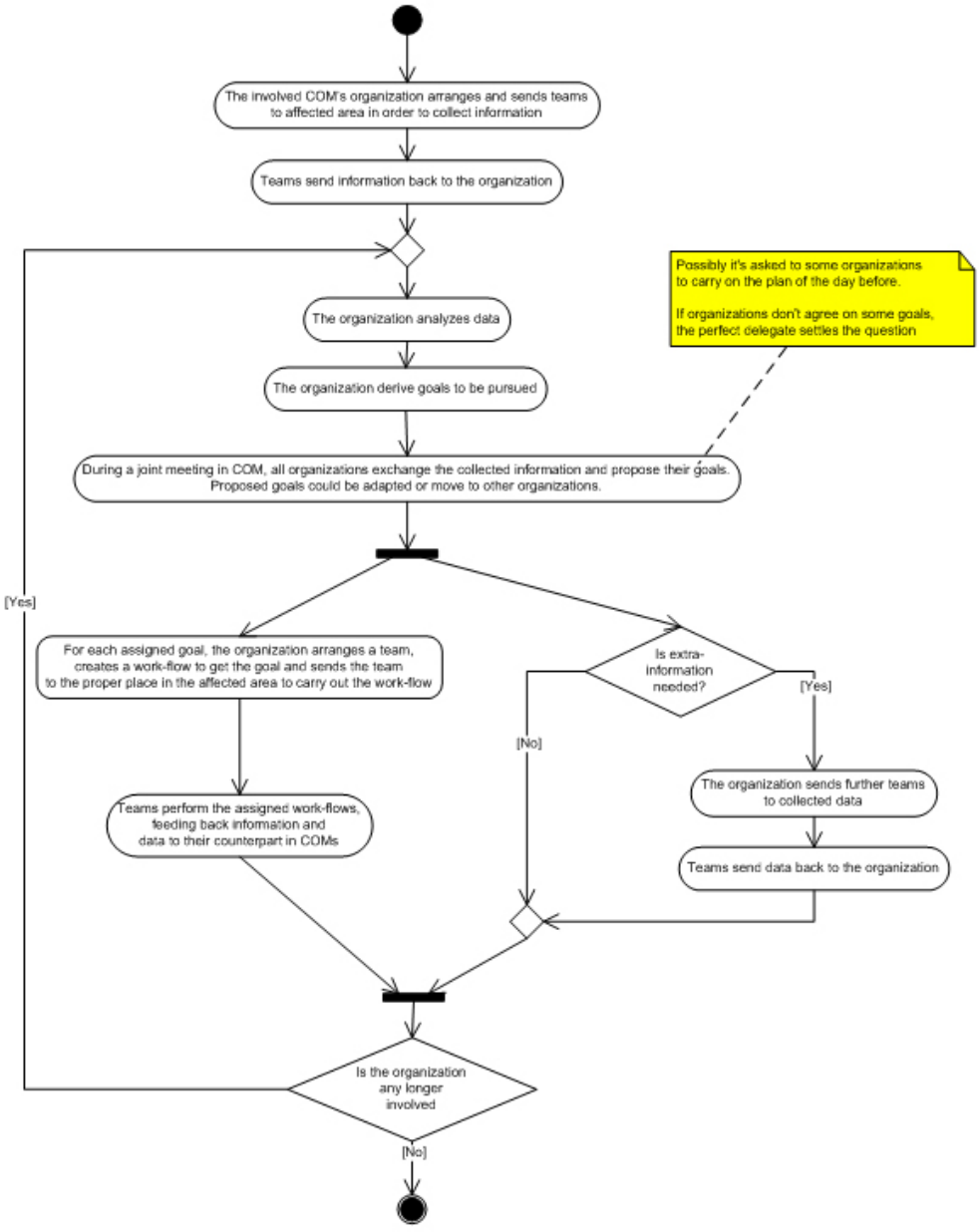


Figure 5.5: Activity Diagram describing the Flood Scenario

The outcomes of the storyboards, HTAs and the plan of execution can be found in the following subsections.

5.4.1 Storyboards and HTA for the Earthquake Scenario

This subsection shows 5 storyboards derived directly from earthquake scenario. Each storyboard is analyzed through HTA, that describes the low-level tasks performed by the actor involved to reach the goal proposed. In order to understand better the analysis carried out, we propose a summary of the earthquake scenario used for obtain storyboards: *"At 10:30 A.M. a violent earthquake of 6 degree on the Richter scale hit the south of Italy, with severe damages in a Calabrian town of 34.000 inhabitants. Furthermore, it is reported that the earthquake has provoked damages to the things and the people in many other Calabrian cities".*

HTA for the Storyboard: "Restoring Essential Services"

- **Actor :** Ferrovie dello Stato (State Railways).
- **Phase :** Response and Short-Term Recovery Phases.
- **Initial State :** The detachment of the State Railways of the city is alerted from the COM established in that zone. The violent earthquake that hit the city has provoked the derailment of three wagons of a passenger train. The number of wounded is not known.
- **Relevant Conditions :** Fire Brigade, Police and Red Cross have been already alerted to intervene and start their operations of first help on the site.
- **Final State :** The emergency situation must have been relieved and the railway service activated again.
- **Main Goal :** Restore the service.
- **Duration :** 6-7 hours.
- **Dependencies :**
  - ***Fire Brigade :*** The duty of the fire-fighters is to evacuate all people from the train.
  - ***Police :*** Policemen secure the area in order to guarantee public safety.
  - ***Red Cross :*** Red Cross operators are prepared to transfer wounded persons to the ambulance.
- **Plan of Execution :**
  - **Plan 0 :** Do 1, 2, 3, 4, 5 in this order.
  - **Plan 2 :** Do 2.1, 2.2, 2.3 in any order.

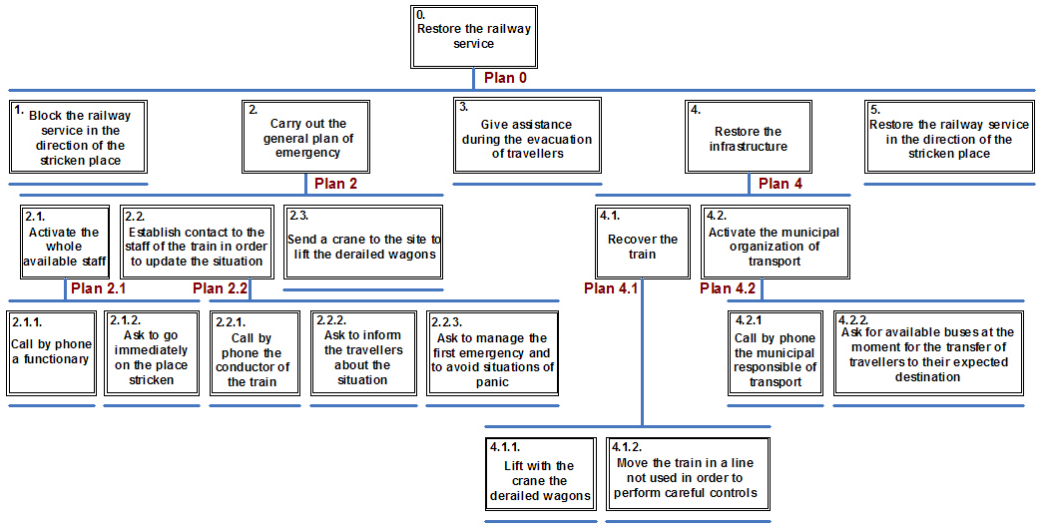


Figure 5.6: HTA for "Restoring Essential Services"

- **Plan 2.1 :** Repeat 2.1.1 and 2.1.2 in this order while all available functionaries haven't been alerted to go on the place stricken.
- **Plan 2.2 :** Do 2.2.1; then do 2.2.2 and 2.2.3 in any order.
- **Plan 4 :** Do 4.1, 4.2 in the same time.
- **Plan 4.1 :** Do 4.1.1, 4.1.2 in this order.
- **Plan 4.2 :** Do 4.2.1, 4.2.2 in this order.

HTA for the Storyboard: "Rescue People out of Debris"

- **Actor :** Fire Brigade.
- **Phase :** Response Phase.
- **Initial State :** The Fire Brigade headquarters of the zone are alerted by the Regional Civil Protection Department. The order is to arrive immediately at a zone 10 Km away from the headquarters with a suitable number of teams since (according to the last news) the collapse of two residences has happened. It seems that some inhabitants have entrapped under the debris; it is not excluded that there are other precarious buildings.
- **Relevant Conditions :** Five ambulances have been alerted to go on the place. Moreover, four teams of police are going to the place too.
- **Final State :** All people trapped under the debris must be rescued.
- **Main Goal :** Save as many human lives as possible.

- **Duration :** 1-2 days.
- **Dependencies :**
  - ***Police :*** Policemen secure the area in order to guarantee public security and to avoid actions of looting.
  - ***Red Cross :*** Operators of the Red Cross are ready to intervene as soon as someone could be identified within the debris.

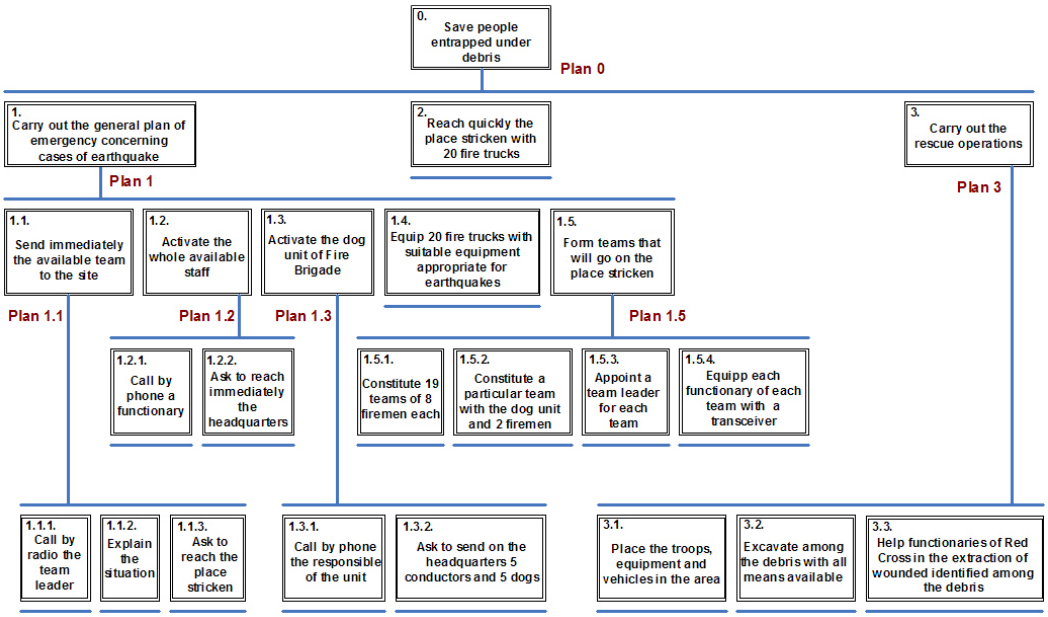


Figure 5.7: HTA for "Rescue People out of Debris"

- **Plan of Execution :**
  - **Plan 0 :** Do 1, 2 in this order. When the fire trucks arrive on the place stricken, do 3.
  - **Plan 1 :** Do 1.1; then do 1.2, 1.3, 1.4 in the same time. Then do 1.5.
  - **Plan 1.1 :** Do 1.1.1, 1.1.2, 1.1.3 in this order.
  - **Plan 1.2 :** Repeat 1.2.1, 1.2.2 in this order while all available functionaries haven't been alerted to reach the headquarters.
  - **Plan 1.3 :** Do 1.3.1, 1.3.2 in this order.
  - **Plan 1.5 :** Do 1.5.1, 1.5.2 in any order. Then do 1.5.3, 1.5.4 in any order.
  - **Plan 3 :** Do 3.1; then do 3.2, 3.3 in the same time.

HTA for the Storyboard: "Establishing a Medical Point"

- **Actor :** Red Cross.
- **Phase :** Short-Term Recovery Phase.
- **Initial State :** In a city district the collapse of two residences is verified. 75 people have been extracted from debris; the serious wounded have been brought in the hospital by the ambulances, while the light wounded have been treated on the site. The Civil Protection Department has decided to build a tent city on the near football field to establish a resort to the homeless people. Moreover it is necessary to establish an Advanced Medical Point on the field to assist all the wounded.
- **Relevant Conditions :** Voluntary Associations and some functionaries of Civil Protection are available at the site, with the purpose of establishing the tent city.
- **Final State :** The Advanced Medical Point must be established in the briefest possible time for assisting serious and light wounded persons.
- **Main Goal :** Establish an Advanced Medical Point in the tent city.
- **Duration :** 2-3 hours.
- **Dependencies :**
  - ***Civil Protection :*** Functionaries of Civil Protection help technicians of Red Cross in building of the Advanced Medical Point.
- **Plan of Execution :**
  - **Plan 0 :** Do 1, 2, 3 in the same time. When all the staff and materials arrive on the place stricken do 4, 5 in this order.
  - **Plan 1 :** Do 1.1, 1.2, 1.3 in any order. Then do 1.4.
  - **Plan 1.1 :** Repeat 1.1.1, 1.1.2, 1.1.3 in this order while there aren't still 5 doctors available to intervene.
  - **Plan 1.2 :** Repeat 1.2.1, 1.2.2, 1.2.3 in this order while there aren't still 10 nurses available to intervene.
  - **Plan 1.3 :** Do 1.3.1, 1.3.2, 1.3.3 in this order.
  - **Plan 2 :** Do 2.1, 2.2, 2.3 in this order.
  - **Plan 4 :** Do 4.1, 4.2, 4.3 in this order.
  - **Plan 4.2 :** Do 4.2.1, 4.2.2, 4.2.3, 4.2.4 in any order.
  - **Plan 5 :** Do 5.1, 5.2 in any order. Then do 5.3.

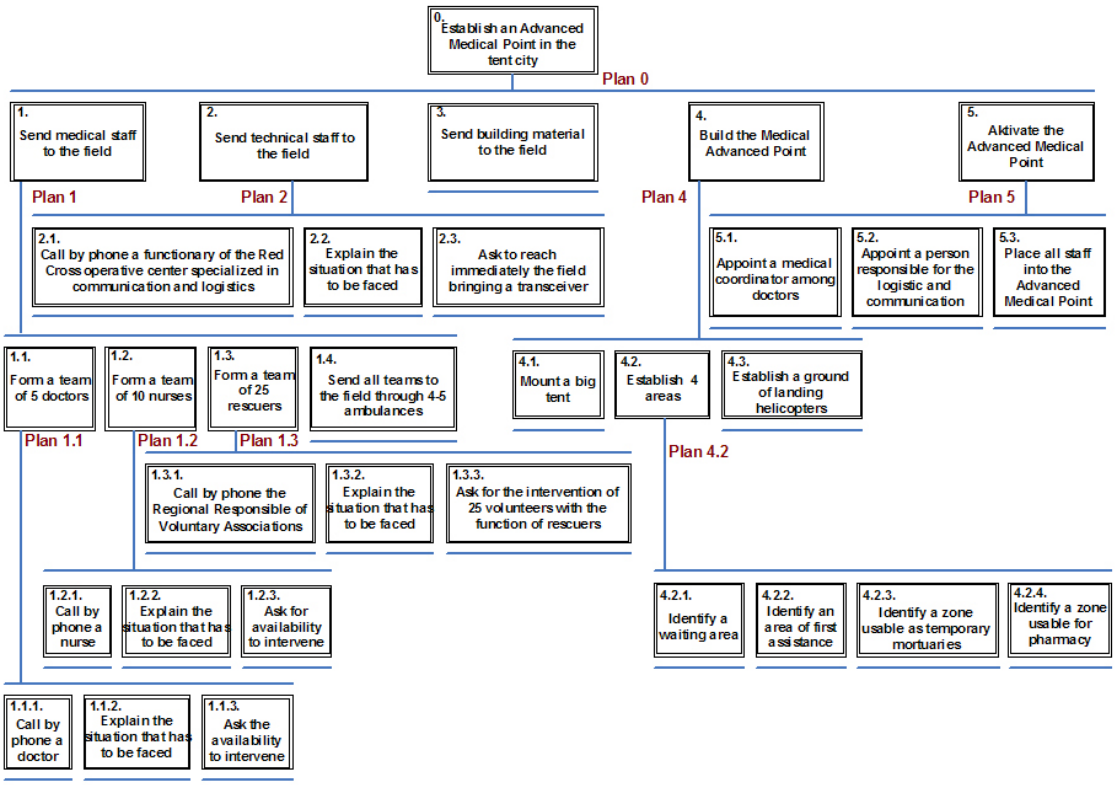


Figure 5.8: HTA for "Establishing a Medical Point"

HTA for the Storyboard: "Transfer Resources for a Tent City"

- **Actor :** Civil Protection.
- **Phase :** Short-Term Recovery Phase.
- **Initial State :** In a city district the collapse of two residences is verified. 75 people have been extracted from debris; the serious wounded have been brought in the hospital by the ambulances, while the light wounded have been treated on the site. The Civil Protection Department has decided to build a tent city on the near football field to establish a resort to the homeless people.
- **Relevant Conditions :** A police patrol is ready to intervene in case of need. Some functionaries of Civil Protection are already on the site to evaluate the situation.
- **Final State :** The tent city must provide tents, electrical power and hygienic services. In addition it would be desirable to have the possibility to guarantee warm meal to homeless people.

- **Main Goal :** Deliver the necessary resources for the construction of the tent city to the site.
- **Duration :** 4-5 hours.
- **Dependencies :**
  - ***Police :*** The order for the policemen is to guarantee public security and to avoid actions of looting.
  - ***Civil Protection :*** Some functionaries of Civil Protection are already on the site to evaluate the situation and to estimate which resources will be necessary for the construction of the tent city.
- **Notes :**
  - Task 2.1.2 is developed in details in storyboard "Establishing a Medical Point".
  - Storyboard "Managing a Tent City" is a follow up of storyboard "Transfer Resources for a Tent City".

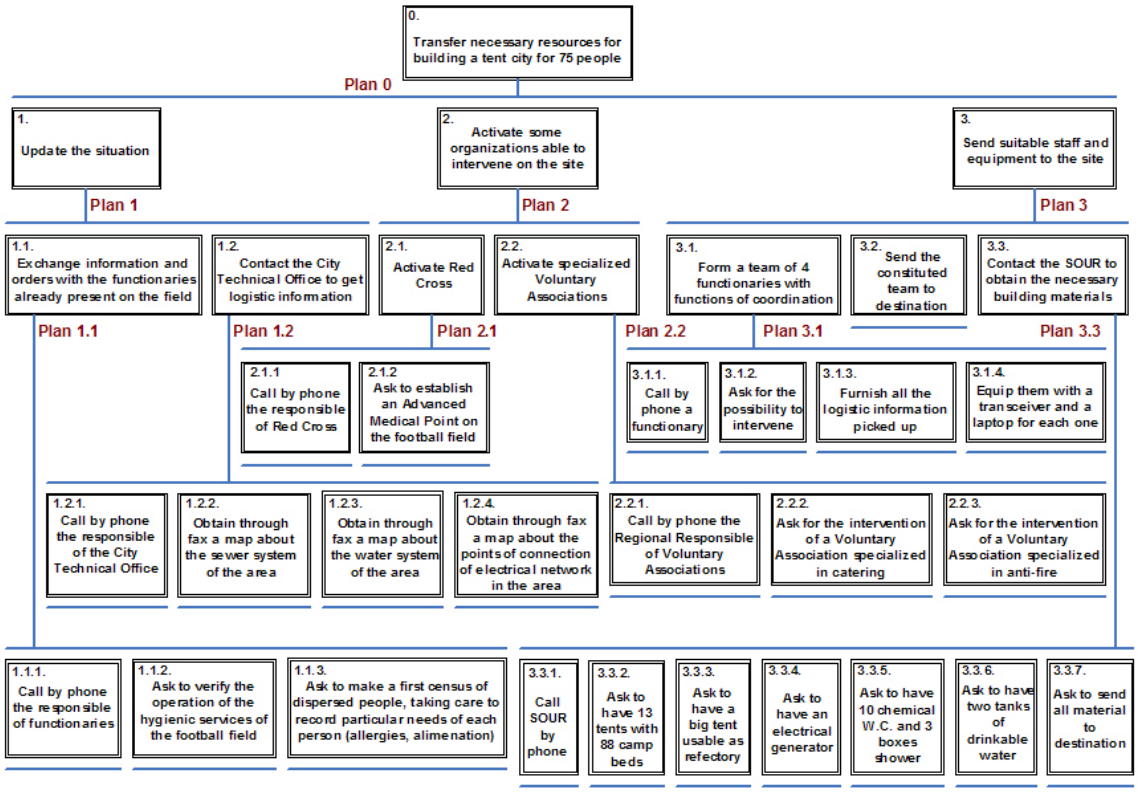


Figure 5.9: HTA for "Transfer Resources for a Tent City"

- **Plan of Execution :**
  - **Plan 0 :** Do 1; then do 2, 3 in the same time.
  - **Plan 1 :** Do 1.1, 1.2 in any order.
  - **Plan 1.1 :** Do 1.1.1; then do 1.1.2, 1.1.3 in any order.
  - **Plan 1.2 :** Do 1.2.1; then do 1.2.2, 1.2.3, 1.2.4 in any order.
  - **Plan 2 :** Do 2.1, 2.2 in any order.
  - **Plan 2.1 :** Do 2.1.1, 2.1.2 in this order.
  - **Plan 2.2 :** Do 2.2.1; then do 2.2.2, 2.2.3 in any order.
  - **Plan 3 :** Do 3.1, 3.2, 3.3 in this order.
  - **Plan 3.1 :** Repeat 3.1.1, 3.1.2 in this order while at least 4 functionaries haven't been alerted. Then do 3.1.3, 3.1.4 in any order.
  - **Plan 3.3 :** Do 3.3.1; then do 3.3.2, 3.3.3, 3.3.4, 3.3.5, 3.3.6 in any order. Then do 3.3.7.

HTA for the Storyboard: "Managing a Tent City"

- **Actor :** Civil Protection.
- **Phase :** Short-Term Recovery Phase.
- **Initial State :** In a city district the collapse of two residences is verified. 75 people have been extracted from debris; the serious wounded have been brought in the hospital by the ambulances, while the light wounded have been treated on the site. The Civil Protection Department has decided to build a tent city on the near football field to establish a resort to the homeless people. 10 functionaries of Civil Protection are present on the site.
- **Relevant Conditions :** The S.O.U.R. has sent a lot of building materials to the site: 13 tents with 88 camp beds, a big tent usable as refectory, an electrical generator, 10 chemical W.C., 3 boxes shower and two tankers of drinkable water. Voluntary Associations specialized in catering and anti-fire have been alerted to intervene. Moreover, the Red Cross will build an Advanced Medical Point on the field.
- **Final State :** The tent city must be finished in the shortest possible time and equipped of a the resources sent to the site.
- **Main Goal :** Manage the building of the tent city.
- **Duration :** 8-10 hours.
- **Dependencies :**

- ***Voluntary Associations :*** Functionaries of specialized Voluntary Associations help functionaries of Civil Protection during the building of the tent city. Moreover, they have a rule of first level in the assistance of citizens in the short-term recovery phase.
- ***Red Cross :*** Technicians of Red Cross, with the assistance of functionaries of Civil Protection and volunteers, build a Medical Advanced Point, that will represent a kind of "field hospital".

- **Notes :**
  - Storyboard "Managing a Tent City" is a follow up of storyboard "Transfer Resources for a Tent City".

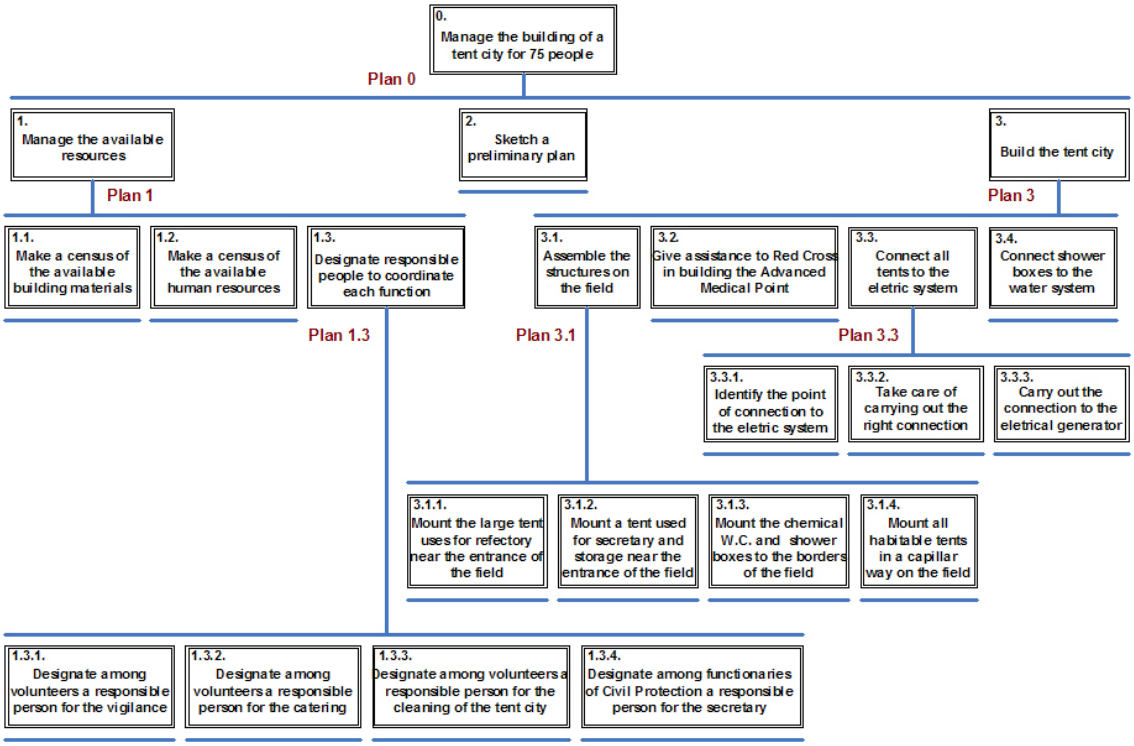


Figure 5.10: HTA for "Managing a Tent City"

- **Plan of Execution :**
  - **Plan 0 :** Do 1, 2, 3 in this order.
  - **Plan 1 :** Do 1.1, 1.2 in any order. Then do 1.3.
  - **Plan 1.3 :** Do 1.3.1, 1.3.2, 1.3.3, 1.3.4 in any order
  - **Plan 3 :** Do 3.1, 3.2 in the same time. Then do 3.3, 3.4 in this order.

- **Plan 3.1 :** Do 3.1.1, 3.1.2, 3.1.3, 3.1.4 in any order.
- **Plan 3.3 :** Do 3.3.1, 3.3.2 in this order. If the electric system doesn't work, do 3.3.3.

5.4.2 Storyboards and HTA for the Flood Scenario

This subsection shows 5 storyboards derived directly from flood scenario. Each storyboard is analyzed through HTA, that describes the low-level tasks performed by the actor involved to reach the goal proposed. In order to understand better the analysis carried out, we propose a summary of the flood scenario used for obtain storyboards: *"During the night a violent and unexpected downpour hit a Calabrian town of 34.000 inhabitants. The town is flooded, which makes the lifesaving operations difficult".*

HTA for the Storyboard: "Restore Railway Service"

- **Actor :** Ferrovie dello Stato (State Railways).
- **Phase :** Response and Short-Term Recovery Phase.
- **Initial State :** The COM which was opened in the catastrophe zone alerts the State Railways of the city. This heavy downpour has provoked an interruption of the electricity in some areas of the city and therefore created problems to the railway practicability. A short-circuit caused fire on a passengers train standing in a gallery.
- **Relevant Conditions :** Fire Brigade, Police and Red Cross have already been alerted to intervene and lead the operations of first help at the operational area. Volunteers of Civil Protection also join them.
- **Final State :** The railway service can again be activated.
- **Main Goal :** Restore the railway service.
- **Duration :** 2-3 hours.
- **Dependencies :**
  - ***Fire Brigade :*** Firemen move with functionaries of State Railways into the gallery. Their task is to extinguish the fire and to evacuate all people out of the train.
  - ***Voluntary Associations :*** The people who have been evacuated are transported out of the gallery by volunteers who afterwards give them assistance.
  - ***Police :*** Policemen secure the area in order to guarantee maintenance of the public security.

– ***Red Cross :*** Red Cross operators move with functionaries of State Railways into the gallery in order to conduct the operations of first help. The ambulances stay outside the gallery.

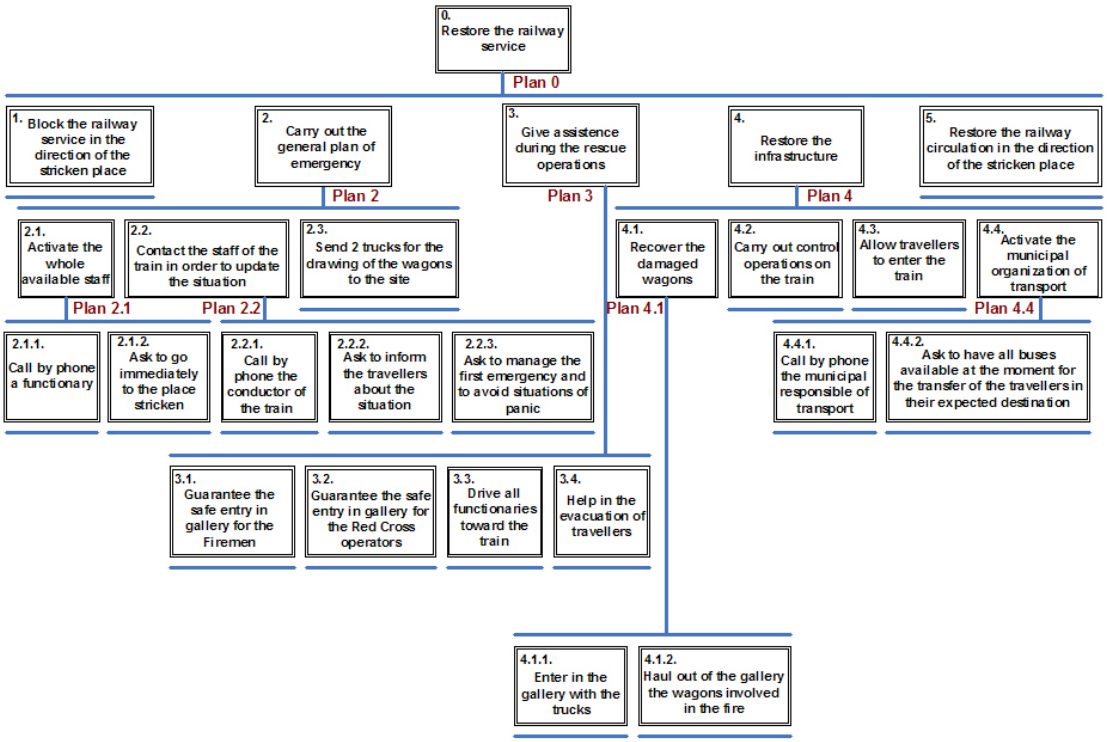


Figure 5.11: HTA for "Restore Railway Service"

- **Plan of Execution :**

  – **Plan 0 :** Do 1, 2, 3, 4, 5 in this order.

  – **Plan 2 :** Do 2.1, 2.2, 2.3 in any order.

  – **Plan 2.1 :** Repeat 2.1.1 and 2.1.2 in this order while all available functionaries haven't been alerted to go on the place stricken.

  – **Plan 2.2 :** Do 2.2.1; then do 2.2.2 and 2.2.3 in any order.

  – **Plan 3 :** Do 3.1 and 3.2 in any order. Then do 3.3 and 3.4 in this order.

  – **Plan 4 :** Do 4.1, 4.2 in this order. Then, if 4.2 has been successful, do 4.3; else do 4.4.

  – **Plan 4.1 :** Do 4.1.1, 4.1.2 in this order.

  – **Plan 4.4 :** Do 4.4.1, 4.4.2 in this order.

HTA for the Storyboard: "Evacuation of People"

- **Actor :** Fire Brigade.
- **Phase :** Response Phase.
- **Initial State :** The Fire Brigade headquarter of the zone is alerted by the Regional Civil Protection Department. In a building of 6 floors, 7 kilometers out of the city centre, water has partially flooded the ground floor (not inhabited). 40 people are captured - women, elderly and children. The number of the wounded people is not yet known.
- **Relevant Conditions :** Two ambulances move to the operational area. Some functionaries of Civil Protection are already at the place to manage the situation. Two police teams have already closed off the area in order to avoid safety problems.
- **Final State :** The building must be evacuated in the shortest possible time in order to rescue all inhabitants.
- **Main Goal :** Rescue all people captured in the building.
- **Duration :** 4-5 hours.
- **Dependencies :**
  - ***Civil Protection :*** At first, functionaries of Civil Protection make a census of inhabitants. After the arrival of the Fire Brigade at the operational area, their order is to give assistance to the already evacuated people.
  - ***Police :*** Policemen have a focus on maintenance of public security and on avoidance of rape.
  - ***Red Cross :*** Functionaries of Red Cross intervene when there are wounded people who have to be helped.
- **Notes :**
  - It is not possible to go into more details concerning Task 3.3 because it is composed of no fixed subtasks. Firemen decide on-site and situation-depending how actions are going to be performed in order to evacuate the people.
- **Plan of Execution :**
  - **Plan 0 :** Do 1, 2 in this order. When the fire trucks arrive on the place stricken, do 3.
  - **Plan 1 :** Do 1.1, 1.2, 1.3 in the same time. Then do 1.4.

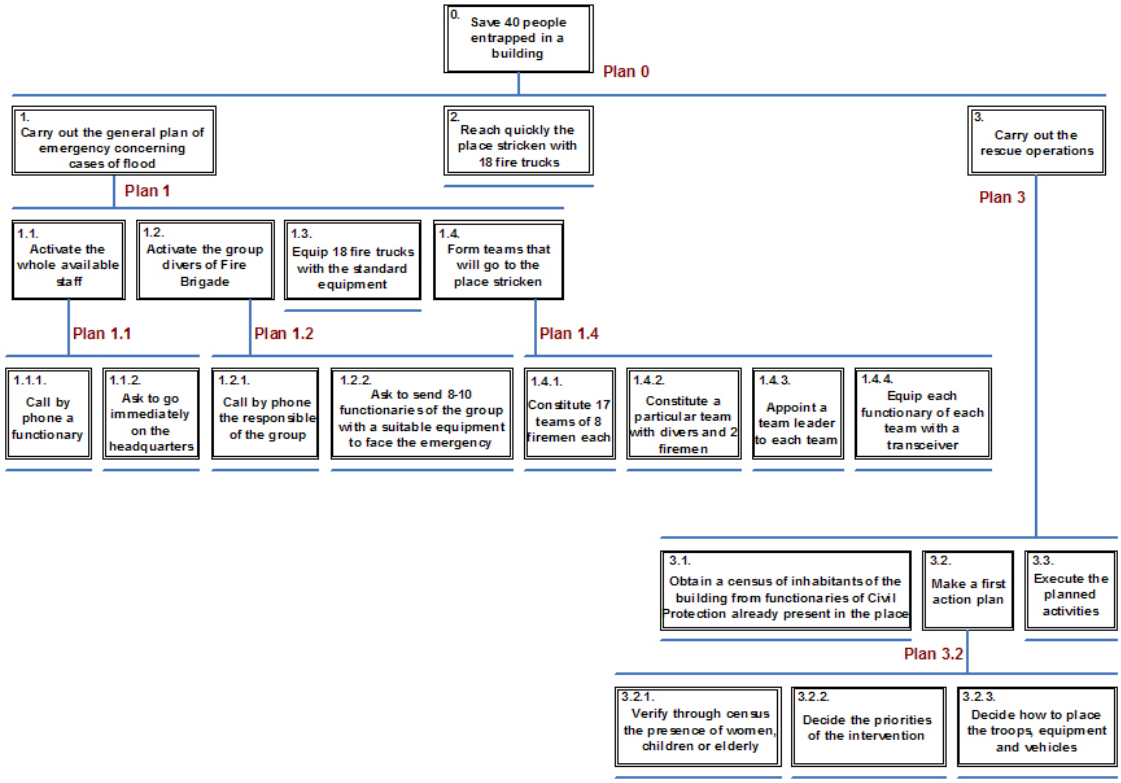


Figure 5.12: HTA for "Evacuation of People"

- **Plan 1.1 :** Repeat 1.1.1, 1.1.2 in this order while all available functionaries haven't been alerted to reach the headquarters.
- **Plan 1.2 :** Do 1.2.1, 1.2.2 in this order.
- **Plan 1.4 :** Do 1.4.1, 1.4.2 in any order. Then do 1.4.3, 1.4.4 in any order.
- **Plan 3 :** Do 3.1, 3.2, 3.3 in this order.
- **Plan 3.2 :** Do 3.2.1, 3.2.2, 3.2.3 in this order.

HTA for the Storyboard: "Assisting during Landslide"

- **Actor :** Red Cross.
- **Phase :** Response Phase.
- **Initial State :** The COM which was opened in the catastrophe zone alerts the Red Cross of the city. The heavy downpour that hit the city has caused a landslide along a city road and stopped the transit in both traffic directions. 5 cars (with 5 people inside each car) have been buried by rubble.

- **Relevant Conditions :** Fire Brigade and Police have already been alerted to intervene. The officers of the Fire Brigade try to remove the rubbles in order to facilitate the intervention of Red Cross. Red Cross is responsible for the wounded people. The Police cares about the traffic management.
- **Final State :** All people have been saved. First help was given to the wounded people and afterwards they have been brought to hospital.
- **Main Goal :** Save as many human lives as possible.
- **Duration :** 2-3 hours.
- **Dependencies :**
  - ***Fire Brigade :*** Firemen try to rescue the wounded people out of the rubble.
  - ***Police :*** Policemen collect and provide information about the current status of the roads and give optimal routing information.
  - ***Person in charge of logistics :*** His duty is the information exchange with the Operative Center of Red Cross about the situation and the reporting to functionaries who operate at the emergency site.
- **Notes :**
  - In Task 4.3.2, the doctor classifies the wounded people. It is the worst case when a wounded person is classified as black code; this usually means that the person has life-threatening deep wounds. The most wounded people are ranked first in getting help.
- **Plan of Execution :**
  - **Plan 0 :** Do 1, 2 in this order. When the responsible arrives on the place stricken do 3 and repeat it so many times as is necessary. When the ambulances arrive on the place stricken do 4.
  - **Plan 1 :** Do 1.1, 1.2, 1.3, 1.4, 1.5 in this order while there aren't still 10 ambulances available to intervene.
  - **Plan 1.4 :** Do 1.4.1, 1.4.2, 1.4.3, 1.4.4 in any order.
  - **Plan 2 :** Do 2.1, 2.2, 2.3 in this order.
  - **Plan 3 :** Do 3.1; then do 3.2, 3.3 in any order.
  - **Plan 4 :** Do 4.1; then do 4.2, 4.3, 4.4, 4.5 in this order while there aren't still wounded to evacuate.
  - **Plan 4.3 :** Do 4.3.1, 4.3.2 in this order.
  - **Plan 4.5 :** Do 4.5.1, 4.5.2 in any order. Then do 4.5.3, 4.5.4 in this order.

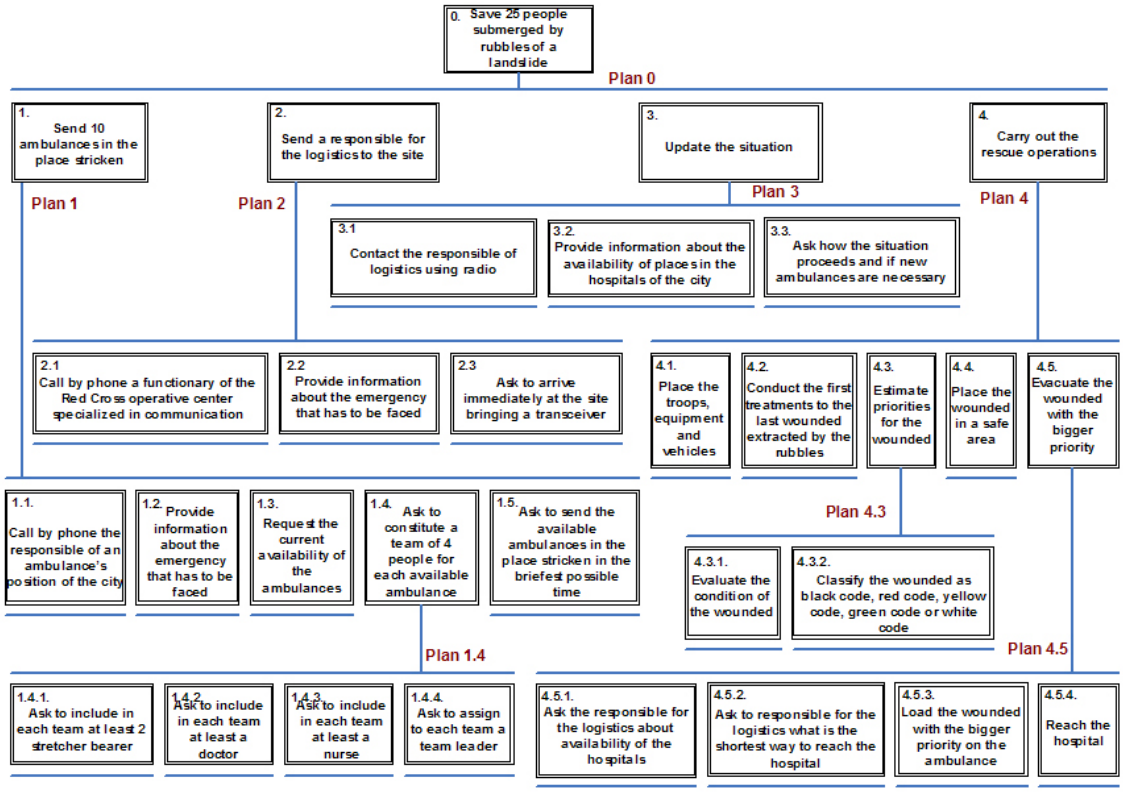


Figure 5.13: HTA for "Assisting during Landslide"

HTA for the Storyboard: "Evacuating Buildings"

- **Actor :** Civil Protection.
- **Phase :** Response Phase.
- **Initial State :** The Regional Civil Protection Department is alerted by the COM opened in the zone. Since the violent downpour doesn't seem to finish, to avoid possibly worse consequences, COM has decided to evacuate the inhabitants of 16 buildings in a district to 5 Km. of distance from your center.
- **Relevant Conditions :** The Officers of the Fire Brigade are arriving at the site to conduct the possible operations of first help. The police will try to maintain the order during the evacuation and to avoid possible actions of looting.
- **Final State :** The buildings must be evacuate in the shortest possible time, securing all the inhabitants.
- **Main Goal :** Evacuate all the people from the buildings.

- **Duration :** 4-5 hours.
- **Dependencies :**
  - ***Fire Brigade :*** The order of the Fire Brigade is to evacuate directly all people from the buildings. The transfer of people to safer sites is managed by Civil Protection.
  - ***Police :*** Policemen secure the area in order to guarantee the maintenance of the public security.
  - ***Voluntary Associations :*** After people have been evacuated, volunteers drive groups of people towards the assembly point identified, giving assistance if necessary. They are coordinated in the actions performed by Civil Protection.

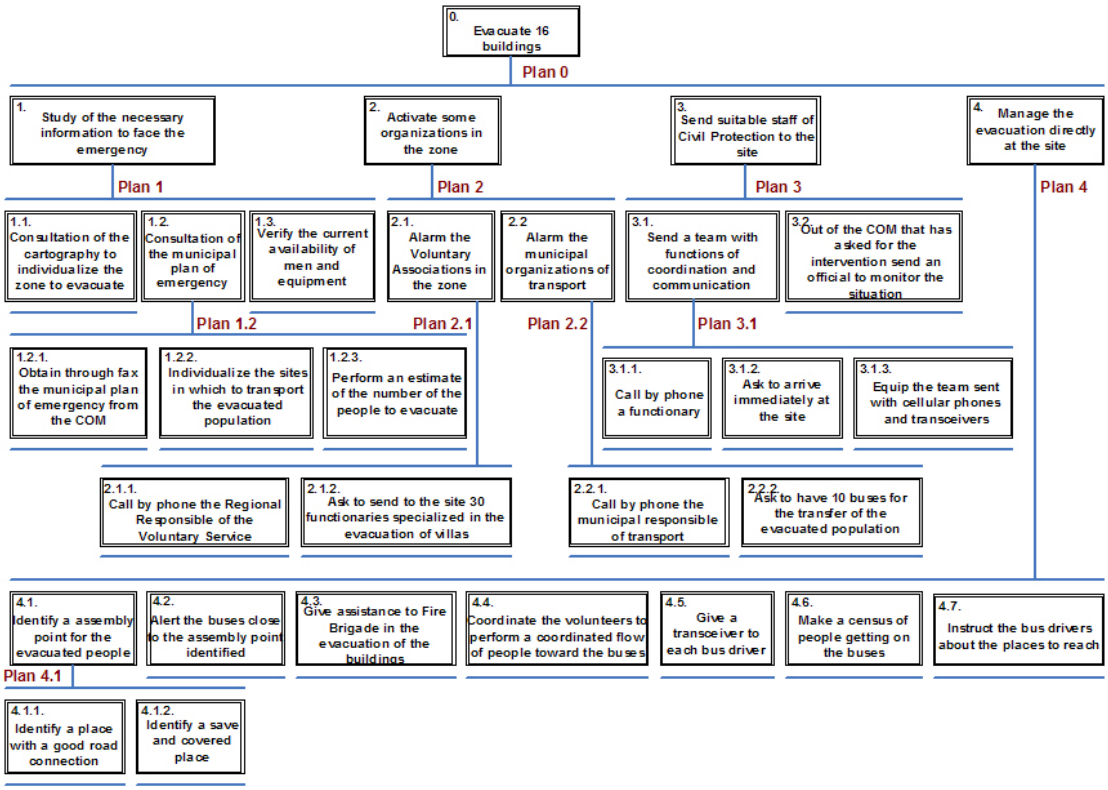


Figure 5.14: HTA for "Evacuating Buildings"

- **Plan of Execution :**
  - **Plan 0 :** Do 1. Then do 2, 3 in the same time. When the Civil Protection staff arrives on the place stricken do 4.
  - **Plan 1 :** Do 1.1, 1.2, 1.3 in any order.

  - **Plan 1.2 :** Do 1.2.1; then do 1.2.2, 1.2.3 in any order.
  - **Plan 2 :** Do 2.1, 2.2 in any order.
  - **Plan 2.1 :** Do 2.1.1, 2.1.2 in this order.
  - **Plan 2.2 :** Do 2.2.1, 2.2.2 in this order.
  - **Plan 3 :** Do 3.1, 3.2 in any order.
  - **Plan 3.1 :** Repeat 3.1.1, 3.1.2 in this order while at least 4 functionaries haven't been alerted to go on the place stricken. Then do 3.1.3.
  - **Plan 4 :** Do 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 in this order.
  - **Plan 4.1 :** Do 4.1.1, 4.1.2 in any order.

HTA for the Storyboard: "Verifying the habitability"

- **Actor :** Civil Protection.
- **Phase :** Response Phase.
- **Initial State :** S.O.U.R. is alerted by the CCS activated in prefecture. According to some notifications of citizens, the violent downpour that hit the city in the night has provoked the collapse of some pillars in a building of 6 floors situated in the city centre. It is necessary to go to the place to verify the habitability (fitness for habitation) of the building.
- **Relevant Conditions :** Since the great distance of the S.O.U.R. from the city affected by the emergency, it would be desirable to involve the Civil Protection Detachment of the city.
- **Final State :** The habitability of the building must be verified in the briefest possible time and the CCS must be informed about the results of the verification.
- **Main Goal :** Verify the habitability of a building.
- **Duration :** 2-3 hours.
- **Dependencies :**
  - ***Civil Protection Detachment of the city :*** It acts directly in the city involved in the emergency. It is coordinated by S.O.U.R.
- **Plan of Execution :**
  - **Plan 0 :** Do 1. When the Civil Protection staff arrives on the place stricken do 2, 3 in this order.
  - **Plan 1 :** Do 1.1, 1.2, 1.3 in this order.

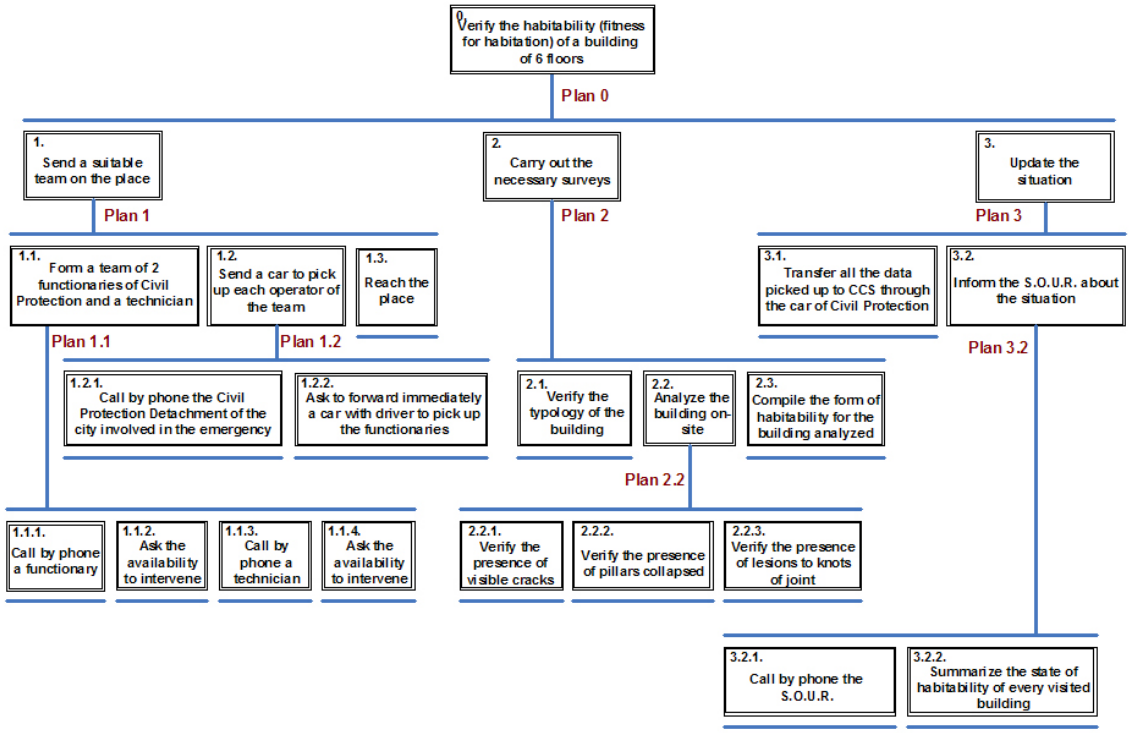


Figure 5.15: HTA for "Verifying the Habitability"

- **Plan 1.1 :** Repeat 1.1.1, 1.1.2 in this order while at least 2 functionaries haven't been alerted to intervene. Then repeat 1.1.3, 1.1.4 in this order while at least a technician hasn't been alerted to intervene.
- **Plan 2 :** Do 2.1, 2.2, 2.3 in this order for the villa in which the collapse has happened. Then repeat these tasks while all the buildings in the immediate proximities have not been verified.
- **Plan 2.2 :** Do 2.2.1, 2.2.2, 2.2.3 in any order.
- **Plan 3 :** Do 3.1, 3.2 in any order.
- **Plan 3.2 :** Do 3.2.1, 3.2.2 in this order.

5.5 User Requirements

As described in Section 4.2 and depicted in Figure 4.3, the requirements were collected with a twofold approach; on the one hand a bottom-up approach (i.e., the Calabrian case study), while on the other hand a top-down approach, such as scenarios (the macro level), storyboards (the medium level), and task analysis (the micro level). User Requirements have been categorized according to forms categories: *general*, *communication*, *Back-End* and *Front-End*. Furthermore, we presented the requirements in a structured way (see Figure 5.16) to provide additional

information for designing, implementing, testing and for requirements tracing issues by specifying: unique identifier, title, an optional description, classification (according to general, communication, Back-End or Front-End), significance for emergency management in general, priority for the implementation, relevancy (for either earthquake or flood scenario or showcase), source, dependency to other requirements and the evaluation method.

5.5.1 User Requirements Listing

This subsection provides the final short-list of user requirements addressed in the WORKPAD project, underlining the identifier and a short description of each requirement.

General (G)

- **G-3** The user must be able to access spatial as well as non-spatial information through one platform.
- **G-5** The user should be able to connect via the WORKPAD system to the European Monitoring and Information Center.
- **G-8** The user must be sure that data confidentially of involved organizations is taken into account.
- **G-11** The user must be able to exploit the WORKPAD system in all kinds of disasters (natural, technical and man-made).
- **G-12** The user must be able to access relevant data-sources of different organizations involved in the emergency management process through WORKPAD.
- **G-14** The way the users interact with the WORKPAD system with respect to the user interface should be exchangeable.
- **G-15** The user must be able to install new WORKPAD nodes (either FE or BE) within 4 hours.
- **G-16** By using the WORKPAD system, the user shall be able to improve the effectiveness in her emergency management processes by 5 - 50%.
- **G-17** The user must be supported in her relevant work-flows in emergency situations by appropriate and adaptive process management techniques within WORKPAD.
- **G-19** In more than 65% of all cases, the workflow management system realized in WORKPAD must improve the management of involved processes.

- **G-20** Decision makers shall be supported through knowledge-based systems integrated in WORKPAD.
- **G-22** The collaboration of FE team members should be improved through the automatic discovery of collaboration patterns and a subsequent appropriate application in the WORKPAD system.
- **G-23** The user should be provided with an overview of the position of teams and equipment through the WORKPAD system.
- **G-24** In case of an emergency, the user must be able to have the WORKPAD system up and running within 30 seconds after the emergency has been confirmed.
- **G-25** Users must be provided with both the PUSH and PULL mode of information provision.
- **G-26** The user must be able to query user-specific and context-sensitive information from the WORKPAD system.
- **G-27** Also users from external organizations (such as voluntary organizations) must be integrable into the WORKPAD system.
- **G-28** The users shall be assisted in collective decision making processes by WORKPAD.
- **G-29** Usability issues shall be taken into account.
- **G-31** The user shall be able to get (quasi) real-time and comprehensive information about the current status of the situation.
- **G-32** By WORKPAD, the user should have the possibility to exploit various integrated, EU related initiatives in order to make use of the expertise available.
- **G-33** The user shall be confronted with a user interface as simple as possible where the number of clicks to reach a certain goal is reduced to a minimum.
- **G-34** The user should be able to get simple assistance such as a help guide or tool-tips.
- **G-35** The user must be notified about events (such as incoming messages, new tasks etc.) in an appropriate way (sound, vibration etc.).
- **G-36** The user should be able to query additional, external data such as, e.g., weather forecasts.
- **G-37** The user shall be supported in her coordination activities by geographic data.

Communication (C)

- **C-1** By using WORKPAD, the user must be able to be connected between different organizations involved in an emergency.
- **C-2** The user must be able to transfer operational data also over low bandwidth networks.
- **C-4** The user's communication must be guaranteed via fault-tolerant network services.
- **C-6** The user must not notice dynamic joins or leaves of network nodes; instead the network must be able to (re-)configure itself.
- **C-8** The user must have an availability rate of more than 99% in time, within the MANET coverage area.
- **C-9** FE team members shall exhibit an average rate of message loss lesser than 10% in the MANET.
- **C-10** When connecting from the FE to the BE, the user shall not encounter an average rate of message loss greater than 10%.
- **C-12** The user shall be able to configure the network concerning the FE and FE/BE link regarding (dynamic, adaptive, and time-varying) QoS in order to define different priorities for different types of traffic.
- **C-13** The user in FE must have an acceptable roundtrip delay when it access the BE by means of the FE-BE link. (Less than 300ms, and in the case of satellite connections, this delay should be lesser than 700ms).
- **C-14** Both the MANET and its connection to the BE must be fully reliable, and thus they should be available to the user even in network challenging situations.

Back-End (B)

- **B-1** The users of the regional Civil Protection Department and of the organizations involved in an emergency must be able to request data stored in the BE.
- **B-2** Authorized users must be able to query diverse relevant information from involved organizations or institutions.
- **B-4** The user must be able to access various data sources integrated in the BE through a well-known interface.
- **B-6** In order to provide the required flexibility, the BE shall use a P2P-based integration of various data sources.

- **B-8** The grid-based query engine of the BE must be scalable.
- **B-9** The query engine in the BE must realize an improvement of 5 - 50% (depending on the specific queries) over a similar traditional system not optimized for emergency (i.e., real time) scenarios.
- **B-10** The BE must enable the integration and exchange of information provided in variety of formats and belonging to different organizations.
- **B-11** Users must be able to get notifications about (generic) information updates at the inter-organizational level related to subscriptions.
- **B-12** The user (e.g., a FE team member) must be supported in her coordinative activities through integrated information coming from BE sources.
- **B-13** The user (e.g., FE team leader) must be able to (re)distribute tasks according to a given process and according to given situations, when relevant events in the BE fire resulting in a change of state of knowledge.
- **B-14** The user must be able to query geographic data from the BE.
- **B-16** The user should be provided with one single administration tool to manage all the functionality of the BE.
- **B-17** The user must be able to define ontologies for the information provided by her system and define mappings to other ontologies.

Front-End (F)

- **F-1** The users in FE teams must be able to electronically communicate with the BE and request data.
- **F-2** The users in FE teams must be able to deliver information to the BE.
- **F-3** The configurations of FE teams must be supported by appropriate communication means (here: establishment of mobile ad hoc networks (MANETs)).
- **F-5** FE entities must be connected through communication paths with other mobile teams for network redundancy reasons.
- **F-6** The user must be able to retrieve data from the BE, to insert new data at the FE, and to transfer and store the data at the BE.
- **F-7** The user must be supported by notification mechanisms.
- **F-9** Information must be presented to the user in an appropriate, user-friendly (i.e. usable) way.

- **F-10** The user must be supported by applications including basic GIS functionalities.
- **F-11** The user should be able to use the WORKPAD system not only on one hardware platform.
- **F-12** The user must be able to use FE devices and software as data input and output devices, necessarily also interacting with the BE.
- **F-13** Users in FE teams or entities shall be localizable.
- **F-17** The user shall be able to install and exploit WORKPAD FE applications on robust hardware.
- **F-18** The user must be able to use WORKPAD applications also in a standalone mode.
- **F-19** In order to increase usability, one single tool should be provided to the user for all necessary tasks.
- **F-20** The users of FE teams should be supported by the WORKPAD system in collaboration, data exchange, and the exploitation of distributed services and information when operating in the field.
- **F-21** The user must be able to communicate with other team members via text messages.
- **F-22** The user should be able to communicate with other team members via audio messages.
- **F-23** The user must be provided with geographic information in the form of raster data enriched with vector data.
- **F-24** The user must be provided with current positions of objects (e.g., vehicles, buildings) or persons (other team members) of interest.
- **F-25** The user must be able to create, modify, or annotate points of interests on a digital map.
- **F-26** The user must be able to distinguish different levels of danger by using different colour codes (e.g., for objects like areas, buildings etc. or also persons in danger).
- **F-27** The user should be able to get a simple routing functionality by choosing to points on a digital map.
- **F-28** The user - when configuring the communication infrastructure - must be able to define workgroups, team member ID, and whether she is coordinator or not.

- **F-29** The user must be able to edit multimedia data such as changing brightness of images etc.
- **F-30** The user should be able to insert and store additional (meta) data regarding multimedia data.
- **F-31** The user must be able to assess and electronically capture the on-site situation on a mobile device.
- **F-32** The user must be able to share file among other members of her team.
- **F-33** The user shall be able to transmit several files to more the one other user at a time.

For reasons of space, this Thesis doesn't report the complete comprehensive-list of User Requirements represented through the Description Form in Figure 5.16. As an instance, Figure 5.17 shows an example of how a User Requirement has been really described in the official documents of the Project.

5.6 Use Case Descriptions

In the WORKPAD requirement engineering methodology, we also adopted the use-case-oriented analysis of requirements. A use case defines an interaction or a sequence of interactions of an actor with the intended system. Hence, a use case defines who (actor) makes what (interactions) with the system to achieve something (goal) without paying attention to the concrete internal details of a system or the implementation of that specific functionality. Actors represent people (or things) that interact in some way with the system. Use cases enable to form a mental model about how an intended system shall work on a conceptual level. It shall embody a common model for all involved stakeholders and in a further step shall assist in capturing requirements. A comprehensive collection and presentation of use cases is central to understanding what the users need from a system (i.e., the requirements) [29].

As depicted in Figure 5.18, User Requirements serves as input for the use cases, and System Requirements are the outputs. Use cases in WORKPAD are described through UML use case diagrams and a tabular notation which contains the following description criteria: ID, Use Case Name, Brief Description, Actors, Preconditions, Final State, Main Flow, Alternatives, Related System Requirements, Related User Requirements, Included Use Cases, Extended Use Cases, Frequency of Execution, Creator, Date, and Last Update. For the use case descriptions is used a standardised template, explained in Figure 4.1.

Figure 5.19 shows the overall use case diagram of the WORKPAD system. The left-hand side of Figure 5.19 shows those actors, team members and team leader, who interact directly with the user interface of the WORKPAD system, while those actors who interact with the Back-End of the WORKPAD system are

on the right hand side. In the middle of the diagram the middleware components of the WORKPAD system, such as the PMS, CMMF and the mobile ad hoc network (MANET) part, are shown. The following subsection describes with major detail the use case concerning the WorkListHandler component.

ID	A unique identifier, composed of a classifier and a sequential number (eg "G-2"): G … General requirement C … Communication requirement B … Back-End requirement F … Front-End requirement
Title	A short title of the requirement giving an overview.
Description (optional)	A more detailed description of the requirement.
Classification	A classification according to: G … General requirement C … Communication requirement B … Back-End requirement F … Front-End requirement
Significance	Depicts the importance of the requirement for an emergency management system in general: Must … This requirement *must* be provided. Shall … This requirement *shall* be provided. Should … This requirement *should* be provided.
Priority	Indicates the priority in terms of an implementation of this requirement within the WORKPAD project: 1 … Mandatory 2 … Desirable 3 … Optional
Relevancy	The requirement is relevant for either scenario 1 or 2 (implying also the storyboards), and the showcase (true/false) (eg "1/false", 'X' would denote "relevant for both")
Source	The requirement was acquired through: U … User analysis (such as interviews, user workshops, HTAs) I … Investigations of related work and/or EU regulations
Dependency (optional)	Indicates a relation between requirements.
Evaluation	The evaluation of this requirement is done via: Ver … Verification: testing (such as software, performance etc.) or review Val … Validation: user/field test, user feedback

Figure 5.16: User Requirements Description Form

ID	G-3
Title	The WORKPAD system must ensure that spatial and non-spatial information can be accessed through one platform.
Description	Nowadays, different information sources are used. Through the WORKPAD system the information access must be homogenised and hence alleviated.
Classification	G
Significance	must
Priority	1
Relevancy	X/true
Source	U
Dependency	--
Evaluation	Ver

Figure 5.17: A User Requirement represented through the Description Form in Figure 5.16

ID	ID of the use case
Use Case Name	Name of the use case
Brief Description	Short description of the use case
Actors	Actors who are involved in the use case
Preconditions	Description of preconditions relevant for the use case
Final State(s)	Final State of the use case
Main Flow	Main flow of the use case
Alternatives	Alternatives for the use case
Related System Requirements	References to system requirements relevant for the use case (see Section 5)
Related User Requirements	References to user requirements relevant for the use case (see Section 5.1)
Included Use Cases	Use Cases which are included by the use case
Extended Use Cases	Use Cases which are extended by the use case
Frequency of Execution	Hardly, sometimes, often, very often
Created by	Name of author
Date created	Date
Last Updated By	Name of author
Date Last Updated	Date

Figure 5.18: Template for Use Case Description

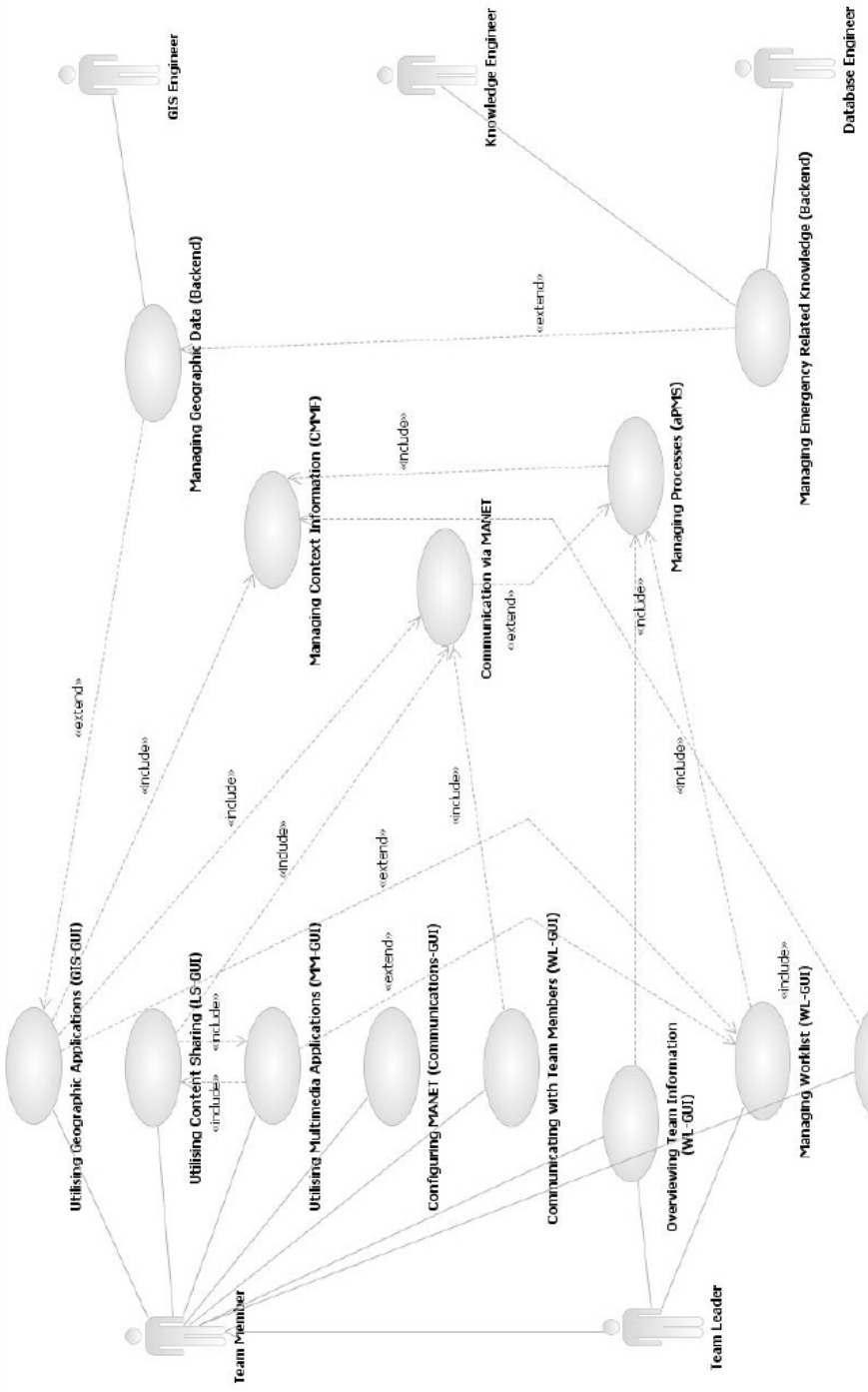


Figure 5.19: Overall Use Case Diagram of WORKPAD

5.6.1 Use Cases of the Worklist Handler component

The Worklist Handler is composed of 3 main components - the "Tasks" component for handling assigned and running tasks, the "Messaging" component for making the audio and text communication accessible over MANET, and finally the "My team" component which provides an overview of the established team and of the team-members' tasks, their status and position. The use case descriptions show how the user interface for the Worklist Handler works. The team leader's PDA has 3 tabs, whereas the PDA of a generic team member has only 2 tabs. Indeed, the generic team member has less functionalities than the team leader. As shown in Figure 5.20, the Worklist Handler Component is directly linked to further WORKPAD subsystems, namely the CMMF, aPMS and MANET. This and the next subsection detail only the Use Case descriptions concerning the main components of the system, the Worklist Handler and aPMS.

5.6.2 Use Cases of the aPMS

The adaptive Process Management System (aPMS) component represents, together with the Worklist Handler, the core of the WORKPAD Front-End. The aPMS is used to adaptively control processes conducted during disaster managements, supporting workflow composition and execution. This component has to manage processes in an adaptive manner based on contextual information and patterns discovered from process mining. The aPMS component cooperates closely with the Worklist Handler, that is intended as a single launcher application which allows team members to receive tasks assigned by the aPMS, and to invoke these tasks, calling user applications required by specific tasks. It is important to highlight that the aPMS component is invisible to the users; in fact they interact only with the WorkList Handler that, in turn, interacts with the aPMS. As shown in Figure 5.20 the aPMS consists of two abstract use cases ('Organising the Team' and 'Organising Tasks') aggregating again several use cases which are described in this subsection.

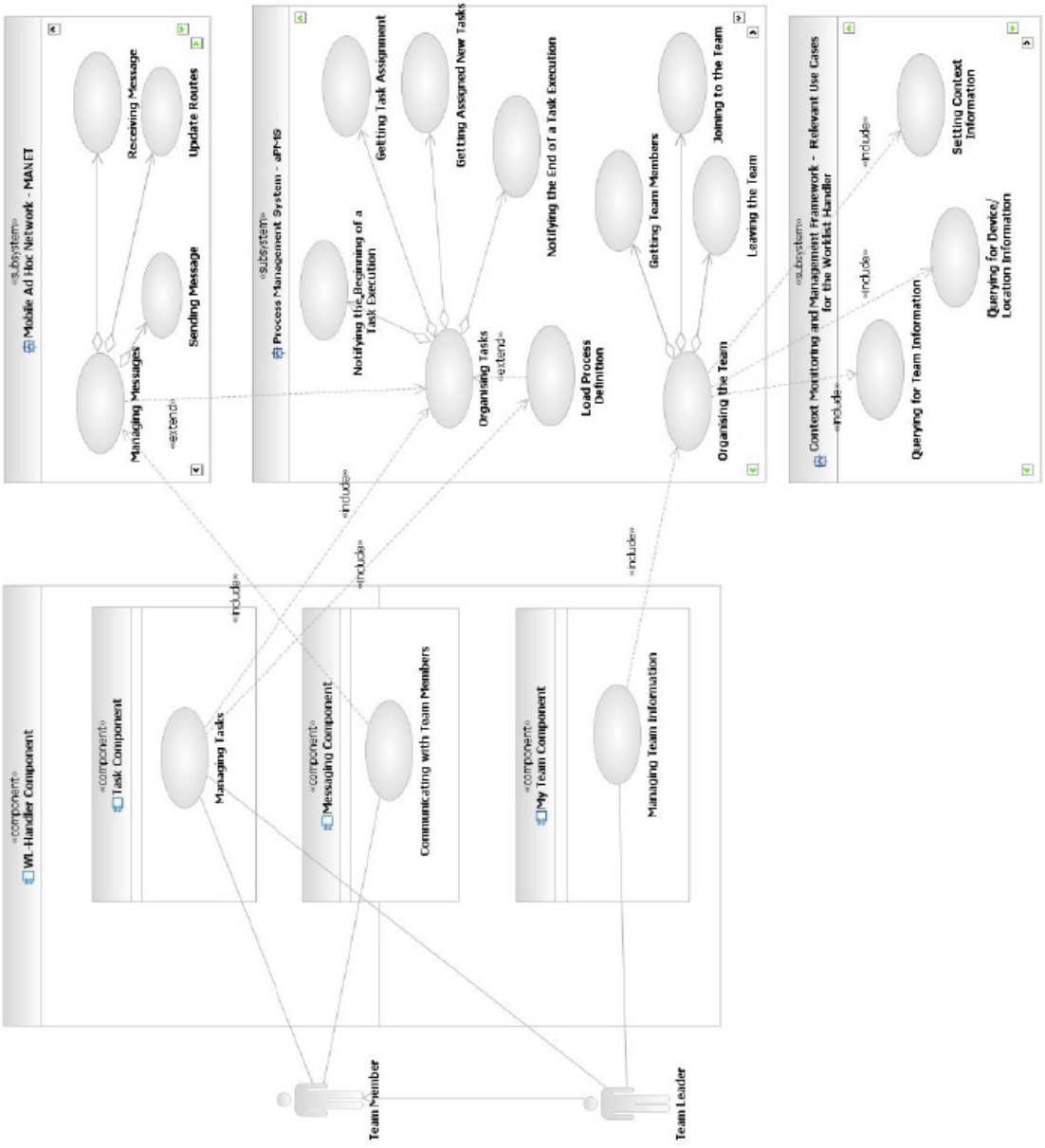


Figure 5.20: Use Case Diagram of the Worklist Handler

ID	UC-WLH-1
Use Case Name	Managing Tasks
Brief Description	Assignment and Execution of a Task.
Actors	Team Member, Team Leader
Preconditions	Skill equipment for that task.
Final State(s)	Application runs and task is fulfilled.
Main Flow	1. After receiving a notification of a new task assignment, the Worklist Handler puts such a task in worklist. 2. The actor picks the task for execution. 3. The system displays information about it. 4. The system invokes respectively runs the appropriate application with the appropriate input data.
Alternatives	None.
Related System Requirements	WH-F-1
Related User Requirements	G-28
Included Use Cases	UC-Abstract-aPMS-1
Extended Use Cases	None.
Frequency of Execution	Very often.
Created by	Andrea Marrella
Date created	13/12/2008
Last Updated By	Andrea Marrella
Date Last Updated	22/02/2008

Figure 5.21: Description of the Use Case "Managing Tasks"

ID	UC-WLH-2
Use Case Name	Communicating with Team Members
Brief Description	Interact with each other by an audio or textual communication.
Actors	Team Member, Team Leader
Preconditions	Communication devices
Final State(s)	Sended audio or textual messages.
Main Flow	1. The actor receives an incoming message. 2. The system displays it. 3. The system broadcasts new message to all team members and also to the team leader. 4. The system displays overview of the received messages.
Alternatives	None.
Related System Requirements	WH-F-2
Related User Requirements	G-28, F-21, F-22
Included Use Cases	UC-Abstract-MANET-1
Extended Use Cases	None.
Frequency of Execution	Very often.
Created by	Andrea Marrella
Date created	13/12/2008
Last Updated By	Andrea Marrella
Date Last Updated	22/02/2008

Figure 5.22: Description of the Use Case "Communicating with Team Members"

5.7 System Requirements

As also argued in [48], System Requirements are derived from various sources :

ID	UC-WLH-3
Use Case Name	Managing Team Informaton
Brief Description	Group of the team members which has to achieve the same goals.
Actors	Team Leader
Preconditions	Actor needs to manage the team.
Final State(s)	Actor will see the member's status.
Main Flow	1. The actor asks for members of her/his team. 2. The system displays team overview. 3. The actor asks for the team member status. 4. The system displays the team member status.
Alternatives	None.
Related System Requirements	WH-F-3
Related User Requirements	G-28
Included Use Cases	UC-Abstract-aPMS-2
Extended Use Cases	None.
Frequency of Execution	Very often.
Created by	Andrea Marrella
Date created	13/12/2008
Last Updated By	Andrea Marrella
Date Last Updated	23/01/2008

Figure 5.23: Description of the Use Case "Managing Team Information"

ID	UC-Abstract-aPMS-1
Use Case Name	Organising Tasks
Brief Description	This use case is an abstract use case allowing to aggregate three sub-use cases ('Notifying the Beginning of a Task Execution', 'Notifying the End of a Task Execution', 'Getting Task Assignment').
Actors	None.
Preconditions	None.
Final State(s)	
Main Flow	None.
Alternatives	None.
Related System Requirements	aPMS-F-1, aPMS-F-2, aPMS-F-3, aPMS-D-1, aPMS-PTS-1, aPMS-PTS-2, aPMS-PTS-3, G-18
Related User Requirements	G-17, G-19, F-20, G-31, G-22, G-16
Included Use Cases	None.
Extended Use Cases	Managing Messages
Frequency of Execution	None.
Created by	Andrea Marrella
Date created	02/02/08
Last Updated By	
Date Last Updated	

Figure 5.24: Description of the abstract Use Case "Organising Tasks"

ID	UC-Abstract-aPMS-2
Use Case Name	Organising the Team
Brief Description	This use case is an abstract use case allowing to aggregate three sub-use cases ('Getting Team Members', 'Joining the Team', 'Leaving the Team').
Actors	None.
Preconditions	None.
Final State(s)	None.
Main Flow	None.
Alternatives	None.
Related System Requirements	aPMS-F-1, aPMS-F-2, aPMS-F-3, aPMS-PTS-1, aPMS-PTS-2
Related User Requirements	F-20, G-31
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	None.
Created by	Andrea Marrella
Date created	02/02/08
Last Updated By	
Date Last Updated	

Figure 5.25: Description of the abstract Use Case "Organising the Team"

ID	UC-aPMS-1
Use Case Name	Joining the Team
Brief Description	This use case allows the actor to join to a team.
Actors	Team Member, Team Leader
Preconditions	The component that is related to the Task List Handler is installed on the FE device and running. The aPMS engine is installed on the FE leader device of the team leader and running. At least a team must to be already created.
Final State(s)	The actor is inserted into a team.
Main Flow	1. The actor calls the functionality 'Joining to the Team' selecting the team (s)he wants to belong. 2. The actor is inserted into the team selected. 3. The aPMS is notified about the insertion of a new actor.
Alternatives	1. The selected team is already full. 1.1 The actor has to choose another team. Otherwise, if all the teams are full, (s)he has to wait for the creation of another team.
Related System Requirements	None.
Related User Requirements	None.
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Sometimes.
Created by	Andrea Marrella
Date created	28/01/2008
Last Updated By	
Date Last Updated	

Figure 5.26: Description of the Use Case "Joining the Team"

ID	UC-aPMS-2
Use Case Name	Leaving the Team
Brief Description	This use case allows the actor to leave the team in which it has worked.
Actors	Team Member, Team Leader
Preconditions	The component that is related to the Task List Handler is installed on the FE device and running. The aPMS engine is installed on the FE leader device and running. At least a team must to be already created. The actor has to belong to a team.
Final State(s)	The actor leaves the team.
Main Flow	1. The actor calls the functionality 'Leaving the Team'. 2. The Task List Handler deletes the actor from the team. 3. The aPMS is notified about the leave of the actor.
Alternatives	None.
Related System Requirements	None.
Related User Requirements	None.
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Sometimes.
Created by	Andrea Marrella
Date created	28/01/2008
Last Updated By	
Date Last Updated	

Figure 5.27: Description of the Use Case "Leaving the Team"

ID	UC-aPMS-3
Use Case Name	Notifying the Beginning of a Task Excecution
Brief Description	It is called when an actor decides to pick a task through the Task List Handler from the list of assigned tasks and start its execution.
Actors	Team Member, Team Leader
Preconditions	The component that is related to the Task List Handler is installed on the FE device and running. The aPMS engine is installed on the leader FE device and running. At least a team must to be already created. The actor has to belong to a team.
Final State(s)	The aPMS is notified about the beginning of a task.
Main Flow	1. The actor picks the task (s)he wants to execute from the list of assigned tasks. 2. The actor calls the functionality 'Start Task'. 3. The Task List Handler notifies the aPMS about the beginning of the selected task.
Alternatives	None.
Related System Requirements	aPMS-F-1
Related User Requirements	F-7
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Very often.
Created by	Andrea Marrella
Date created	29/01/2008
Last Updated By	
Date Last Updated	

Figure 5.28: Description of the Use Case "Notify the beginning of a task execution"

ID	UC-aPMS-4
Use Case Name	Notifying the End of a Task Execution
Brief Description	It is executed when an actor concludes the execution of a task and wants to notify that to the aPMS.
Actors	Team Member, Team Leader
Preconditions	The component that is related to the Task List Handler is installed on the FE device and running. The aPMS engine is installed on the leader FE device and running. At least a team must to be already created. The actor has to belong to a team. At least a task has to be assigned to the actor.
Final State(s)	The aPMS is notified about the ending of a task.
Main Flow	1. The actor calls the functionality 'End Task'. 2. The Task List Handler notifies the aPMS about the ending of the task.
Alternatives	None.
Related System Requirements	aPMS-F-1
Related User Requirements	F-7
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Very often.
Created by	Andrea Marrella
Date created	29/01/2008
Last Updated By	
Date Last Updated	

Figure 5.29: Description of the Use Case "Notify the end of a task execution"

ID	UC-aPMS-5
Use Case Name	Getting Team Members
Brief Description	The Task List Handler calls this functionality when interested in knowing the team composition.
Actors	Team Leader
Preconditions	The component that is related to the Task List Handler is installed on the FE device and running. The aPMS engine is installed on the leader FE device and running. At least a team must to be already created. The actor has to belong to a team.
Final State(s)	The Task List Handler is notified about the team composition.
Main Flow	1. The actor calls the functionality 'Get Team Members'. 2. The aPMS notifies the Task List Handler about the team composition.
Alternatives	None.
Related System Requirements	aPMS-F-1
Related User Requirements	
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Often.
Created by	Andrea Marrella
Date created	30/01/2008
Last Updated By	
Date Last Updated	

Figure 5.30: Description of the Use Case "Getting Team Members"

ID	UC-aPMS-6
Use Case Name	Getting Task Assignment
Brief Description	The Task List Handler calls this functionality when it is interested in knowing the task assignment.
Actors	Team Leader
Preconditions	The component that is related to the Task List Handler is installed on the FE device and running. The aPMS engine is installed on the leader FE device and running. At least a team must to be already created. The actor has to belong to a team.
Final State(s)	The Task List Handler is notified about the task assignment.
Main Flow	1. The actor calls the functionality 'Get Task Assignment'. 2. The aPMS notifies the Task List Handler about the tasks assigned to each team member.
Alternatives	None.
Related System Requirements	aPMS-F-1
Related User Requirements	
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Often.
Created by	Andrea Marrella
Date created	30/01/2008
Last Updated By	
Date Last Updated	

Figure 5.31: Description of the Use Case "Getting Task Assignment"

ID	UC-aPMS-7
Use Case Name	Getting Assigned New Tasks
Brief Description	With this functionality, the leader/member of a team, through the Work List Handler, is notified by the aPMS about the assignment of a new task to be executed. This functionality is executed by aPMS also when adaption is needed in order to fulfil a particular situation (i.e. a disconnection of a team member)
Actors	Team Member, Team Leader
Preconditions	The component that is related to the Work List Handler is installed on the FE device and running. The aPMS engine is installed on the leader FE device and running. At least a team must to be already created. The actor has to belong to a team.
Final State(s)	The actor is notified about the the assignment of a new task to be executed.
Main Flow	1. The aPMS notifies the Work List Handler of the team leader/member about the assignment of a new task to be executed.
Alternatives	None.
Related System Requirements	aPMS-F-1, aPMS-F-3
Related User Requirements	G-17, G-19, G-20, G-28, F-7
Included Use Cases	None.
Extended Use Cases	None.
Frequency of Execution	Very often.
Created by	Andrea Marrella
Date created	29/01/2008
Last Updated By	Andrea Marrella
Date Last Updated	23/02/2008

Figure 5.32: Description of the Use Case "Getting Assigned new Task"

ID	UC-aPMS-8
Use Case Name	Load Process Definition
Brief Description	It is called when the Team Leader decides to load a new process into the aPMS.
Actors	Team Leader
Preconditions	The component that is related to the Work List Handler is installed on the FE device and running. The aPMS engine is installed on the leader FE device and running.
Final State(s)	The new process to be executed is loaded into the aPMS, that starts to execute it, assigning tasks to members without the necessity of any intervention of the team leader.
Main Flow	1. The actor calls the functionality 'Load Process Definition' selecting the process (s)he wants to load into the aPMS. 2. The aPMS loads the process managing automatically its execution.
Alternatives	None.
Related System Requirements	aPMS-F-1, aPMS-F-2, aPMS-PTS-3
Related User Requirements	None.
Included Use Cases	None.
Extended Use Cases	UC-Abstract-aPMS-1
Frequency of Execution	Sometimes.
Created By	Andrea Marrella
Date Created	28/01/2008
Last Updated By	
Date Last Updated	
Remark	The use case was renamed from UC-PM-1 to UC-aPMS-8 and was shifted to the aPMS-component.

Figure 5.33: Description of the Use Case "Load Process Definition"

1. The users of a system;

2. Insights from similar initiatives (i.e., related work);

3. Domain experts or system engineers (i.e., technicians).

(1) and (2) have been covered by the works conducted during the establishment of the twofold approach: bottom-up (i.e., the Calabrian case study, cfr. Section 2.1 and Chapter 5) and top-down (i.e., the examination of relevant research projects and EU legislation and initiatives, cfr. Sections 3.3 and 3.4). Both served by way of the defined user requirements as a source of input for the system requirements generation. Furthermore, the third important source (domain experts or system engineers) is covered by the technical partners in the WORKPAD consortium. To make the expertise and experience of the relevant people explicit use cases were considered as a appropriate tool. The resulting System Requirements are clustered according to the functional entities of the intended WORKPAD system and are listed in Section 5.7.1. Also for the System Requirement presentation we are using a tabular schema (see Figure 5.34) that offers the following description criteria: ID, Title, Brief Description, Classification, Significance, Priority, Dependency, and Evaluation.

The chosen tabular notations offer the possibility to show the interrelations between the various stages of the adopted methodology.

ID	ID of the requirement
Title	A short title of the requirement giving an overview.
Brief Description	A more detailed description of the requirement.
Classification	**D**-Data Related Requirement, **F**-Function/Feature Related Requirement, **U**-Usability Requirement, **PTS**-Performance/Throughput/Security Requirement
Significance	**Must** (This requirement must be provided.), **Shall** (This requirement shall be provided.), **Should** (This requirement should be provided.)
Priority	Indicates the priority in terms of an implementation of this requirement within the WORKPAD project: **1**-Mandatory, **2**-Desirable, 3-Optional
Dependency	Indicates a relation between requirements.
Evaluation	The evaluation of this requirement is done via Verification (**Ver**: system testing or system/software review) or Validation (**Val**): user/field test, user feedback).

Figure 5.34: System Requirements Description Form

5.7.1 System Requirements Listing

This subsection provides the final short-list of System Requirements addressed in the WORKPAD project, underlining the identifier and a short description of each requirement.

Worklist Handler (WH)

- **WH-F-1** Applications must provide interfaces for the Worklist Handler component.
- **WH-F-2** The Worklist Handler component must be able to support DOM-based processing of documents.
- **WH-F-3** The Worklist Handler component must provide an aPMS interface.

Context Monitoring and Management Framework (CMMF)

- **CMMF-F-1** Web service middleware connectivity.
- **CMMF-F-2** Transparent placement of queries and subscriptions.
- **CMMF-F-3** Context can be read and manipulated by the team members.
- **CMMF-F-4** System must be able to detect and remove redundant information.

- **CMMF-F-5** System must be able to detect and remove conflicting information.
- **CMMF-F-6** System must be able to merge information.

The MANET Management (MANET)

- **MANET-F-1** WORKPAD devices must be able to set up WIFI connections with other devices in the team.
- **MANET-F-2** The MANET component must be able to support different Windows Platforms running on the WORKPAD devices.
- **MANET-F-3** The MANET component allows the WORKPAD applications to be used in a standalone mode.
- **MANET-F-4** The MANET component must be able to assign unique numbers within a workgroup.
- **MANET-F-5** A MANET device must contain robust hardware.
- **MANET-F-6** The MANET component shall provide communication mechanisms to higher level user applications to collaborate, exchange data, and make use of services and information when operating in the field.
- **MANET-F-7** The MANET component must provide a high network availability for the WORKPAD applications.
- **MANET-PTS-1** The MANET component should be energy efficient.
- **MANET-PTS-2** The MANET component shall provide secure wireless communication, intrusion detection, cryptography, and protocol verification.
- **MANET-PTS-4** The MANET component should provide a low delay connection between peers in the Front End.
- **MANET-PTS-5** The MANET component must provide a high capacity connection between peers in the Front End.

Adaptive Process Management System (aPMS)

- **aPMS-F-1** The aPMS component must be able to communicate with the Work List Handler of team members (including the team leader).
- **aPMS-F-2** The aPMS component should provide an XML transformer.
- **aPMS-F-3** The aPMS component must be able to assign tasks without any intervention of the team leader.

- **aPMS-D-1** The data provided by the aPMS should be described in XML format.
- **aPMS-PTS-1** When a process is loaded into the aPMS engine, it shall start the assignation of the tasks after not more than 10 seconds.
- **aPMS-PTS-2** The aPMS component, as the whole P2P network, must be capable to function even if some nodes of the network are not available.
- **aPMS-PTS-3** Processes should not remain stalled for more than 30 seconds.

GIS Front-End (GIS-FE)

- **GIS-FE-F-1** The GIS-FE component must be able to group geographic information to layers.
- **GIS-FE-F-2** The GIS-FE component must provide the possibility to show or hide the geographic information.
- **GIS-FE-F-3** The GIS-FE component must be able to distribute geographic information to all reachable peers belonging to one peer group.
- **GIS-FE-F-4** One peer in the peer group must be able to persist data into the BE - possibly via a multi-hop path.
- **GIS-FE-F-5** Connectivity between the peers within one peer group running the GIS FE application must be establishable.
- **GIS-FE-F-6** The GIS-FE component must provide the possibility to create and annotate specific points-of-interests (POI).
- **GIS-FE-F-7** The GIS-FE component must provide appropriate interfaces to interact with the underlying data model to add/delete/change geodata (i.e., objects).
- **GIS-FE-F-8** The geography application should provide a minimum of three different pre-defined zoom levels.
- **GIS-FE-PTS-1** The GIS-FE component must provide means to distribute data within the peer group.
- **GIS-FE-PTS-2** The GIS-FE component shall provide techniques to deal with steady disconnections.
- **GIS-FE-PTS-3** A consistent data view (among all peers in one peer group) shall be established after 5 seconds after an update.
- **GIS-FE-PTS-4** The GIS-FE component must conform to the accuracy provided by the used GNSS.

- **GIS-FE-PTS-5** The GIS-FE component must provide an update of positions every second.
- **GIS-FE-U-1** The GIS-FE component must be usable with a stylus.
- **GIS-FE-U-2** The GIS-FE component must provide standard map interaction functions such as pan, zoom, rotate, and load map.
- **GIS-FE-U-3** The GIS-FE component should additionally be usable by touching the screen without any extra device.
- **GIS-FE-U-4** The GIS-FE component must be able to present map-based, geo-feature based and text data on the screen of a portable device.
- **GIS-FE-U-5** The GIS-FE component should be able to present geo-referenced photos.
- **GIS-FE-U-6** The geography application should display relevant longitude and latitude coordinates, a scale, and the orientation.
- **GIS-FE-D-1** The GIS-FE component must be able to model and process map-based, geo-feature-based and text data on the screen of a portable device.
- **GIS-FE-D-2** The GIS-FE component should be able to provide access to geo-referenced photos.
- **GIS-FE-D-3** The GIS-FE component must be able to interface to and integrate context data.
- **GIS-FE-D-4** The GIS-FE component must use a data model approach that allows querying context data, distributing data to other peers over a MANET, and rendering it on the screen of a portable device.

Lightweight Storage (LS)

- **LS-F-1** The LS component must provide an interface to configure a peer being part of a group.
- **LS-F-2** The LS component has to be able to communicate with other peers within the group.
- **LS-F-3** The LS component shall provide functionalities in order to put new files in shared folder.
- **LS-F-4** The LS component must implement a way to send shared files to peers within the group.

- **LS-F-5** The LS component must provide a search functionality in order to find useful data in other peers' shared folders.
- **LS-F-6** The LS component must offer the opportunity to download files coming from a search.
- **LS-F-7** The LS component shall notify other peers of any change on its shared folder.
- **LS-F-8** The LS component should provide a connected peers browsing functionality.
- **LS-D-1** The LS component should provide a folder to store shared files, e.g. files downloaded from other peers or downloadable from them.

Multimedia Editor (MME)

- **MME-F-1** The MME component must be able to present images and videos in an appropriate way.
- **MME-F-2** The multimedia application must be able to provide functionalities such as open, save, close, delete, and rename a multimedia file; in addition further functionalities such as cropping, resizing, zooming, changing brightness, and rotating must be provided.
- **MME-D-1** The MME component must provide interfaces to the LS component.
- **MME-D-2** The MME component must support standard image formats.
- **MME-D-3** The device must have filesystem.

Context Editor (CE)

- **CE-F-1** The Context Editor must be able to support DOM-based processing of documents.
- **CE-F-2** The Context Editor must support XML HttpRequests.
- **CE-F-3** The Context Editor must be able to execute XPATH queries.
- **CE-D-1** The Context Editor must provide interfaces to the CMMF component.

GIS Back-End (GIS-BE)

- **GIS-BE-F-1** The GIS-BE component must provide means to exchange and configure different GIS servers (GIS datasources).
- **GIS-BE-F-2** The GIS-BE component must provide means to retrieve geographic features.
- **GIS-BE-F-3** The GIS-BE component must provide means to store geographic features.
- **GIS-BE-F-4** The GIS-BE component should be able to provide notifications about changes in geographic features.
- **GIS-BE-F-5** The GIS-BE component must uniquely identify geographic features.
- **GIS-BE-F-6** The GIS-BE component must provide means to configure the GIS datasources to use.
- **GIS-BE-F-7** The GIS-BE component must be able to integrate heterogenous GIS datasources on a semantic level.
- **GIS-BE-F-8** The GIS-BE component must be able to uniquely identify different features from different sources. The system must provide a mapping to these sources.
- **GIS-BE-F-9** The GIS-BE component must provide means to retrieve geographic maps.

P2P Data Integration (DI)

- **DI-PTS-1** The P2P component should guarantee autonomy and flexibility of participants of a P2P network.
- **DI-PTS-2** The P2P component, as the whole p2p network, must be capable to function even if some nodes of the network are not available.
- **DI-PTS-3** The scalability of the P2P DI component should not be highly affected by the number of peers.
- **DI-PTS-4** The query engine must realize improvement with respect to traditional systems.
- **DI-PTS-5** The P2P component should have secure access to the BE.
- **DI-PTS-6** The P2P component must be available for the period of the emergency (ideally 24x7).

- **DI-PTS-7** The P2P component must allow concurrent access form clients to the query engine.
- **DI-PTS-8** The P2P component should preserve data integrity.
- **DI-F-1** The P2P component must be able to integrate heterogeneous data sources and provide them through a uniform interface.
- **DI-F-2** The P2P component must allow meaningful queries.
- **DI-F-3** The P2P component should support semantic query mechanism.
- **DI-F-4** The P2P component should support complex FE workflows by exposing a query interface over invoked data.
- **DI-F-5** The P2P component must support notification of information updates to interested parties.
- **DI-U-1** The P2P component must allow rapid configuration of a new node entering the network.
- **DI-U-2** The P2P component should offer a graphical interface to model knowledge through ontologies.
- **DI-U-3** The P2P component should aid the user to define sources.
- **DI-U-4** In order to facilitate the peer's sources definition the system should permit the user to introspect the sources' schemas.
- **DI-D-1** The participants of the BE P2P network must be able to request data stored in the various peers of the BE.
- **DI-D-2** The P2P component must be able to provide data of interest to the participant organizations.

For reasons of space, this Thesis doesn't report the complete comprehensive-list of System Requirements represented through the Description Form in Figure 5.34. As an instance, Figure 5.35 shows an example of how a System Requirement has been really described in the official documents of the Project.

ID	WH-F-1
Title	Applications must provide interfaces for the Worklist Handler component.
Brief Description	Some applications, which use capabilites of the Worklist Handler must provide interfaces for the Worklist Handler component.
Classification	F
Significance	Must
Priority	1
Dependency	None.
Evaluation	Ver

Figure 5.35: A User Requirement represented through the Description Form in Figure 5.34

Chapter 6

Prototypes Artefacts

The creation of a validation and evaluation plan (see Section 4.3) has allowed to carry out an effective assessment of the project outcomes. This was necessary for evaluating and validating the project activities and corresponding results from several distinct perspectives such as individual technological components, as well as the entire system being developed in the project. This Chapter aims to describe in depth the various types of user tests designed and executed to assess the WORKPAD system, giving details about their results, conclusions and recommendations.

The Chapter is organized as follow:

- Section 6.1 describes the performance of a mock-up test with the different WORKPAD components. The test was performed with an on-line questionnaire and gave a first insight of how users evaluated the different single component mock-ups.

- Section 6.2 presents the performance of controlled experiments about some selected components of the WORKPAD system, executed in a lab environment under controlled conditions.

- Section 6.3 details a test performed with Cooperative Evaluation method.

- Section 6.4 shows a usability test carried out with non-expert users (people outside of the domain of emergency management).

- Section 6.5 describes the results of a simulation performed in a realistic setting with emergency operators, without the support of the WORKPAD system.

- Section 6.6 presents the design, the execution and the outcomes of the final showcase of the project.

6.1 Online Pre-Tests

Referring to the WORKPAD validation plan (see Figure 4.6), we performed a pre-test as soon as mock-ups of the components were available and ready for testing with potential users. The main goal of this test was to get a first insight about how usable and understandable the first WORKPAD mock-up was to the users. Moreover, it was important to get feedback from the users, whether the requirements were understood correctly and whether the main functionalities meet the users' demands. The evaluation results were used as improvement recommendations for the developers.

6.1.1 Execution Details

The realization of the pre-test was done using an online questionnaire. Moreover, all Front-End mock-ups were developed in digital form and therefore predestined for online evaluation. As an example, a couple of screenshots representing the online questionnaire are shown in Figures 6.1 and 6.2.

Ten to fifteen test users were expected to test the WORKPAD mock-ups. Each partner developing a component with a graphical user interface (GUI) in the Front-End provided his mock-up in presentation form following certain design guidelines. Then the mock-ups have been harmonized and questions concerning the usability of the mock-up components (i.e., task management, map overview, connection establishment, multimedia and context editor, file share) were developed. As the final users (Calabrian Civil Protection) come from Italy and WORKPAD follows the User-Centered Design approach, the questionnaire and the mock-ups were translated into Italian in order to enable the users to get an impression of the mock-ups and mention improvements of the different WORKPAD components. The mock-up presentation was included in the online questionnaire and is accompanied by several questions. In order to be sure that the questionnaire was consistent and understandable to several project external people, WORKPAD partners performed the online questionnaire before that the final test version was sent to the users in Calabria.

The Calabrian Civil Protection Department, user partner in WORKPAD, was responsible for recruiting the test users. Thirteen people participated in the first pre-test. Ten users were from the Calabrian Civil Protection Department, one from ANAS S.p.A., one from the Department of National Civil Protection and one from KRONOS - Calabria/Accademia Kronos. The users testing mock-ups were selected taking into account their knowledge about the Emergency Management context; in fact, within their organisation, they performed activities such as processing meteorological data, processing climate data, responsible for management activities related to National roads in the Calabria Region, responsible for the Coordination Centre in the Risk and Crisis Management Sector, Civil Protection technical operator, logistic expert, hydrogeological risk monitoring tasks, coordi-

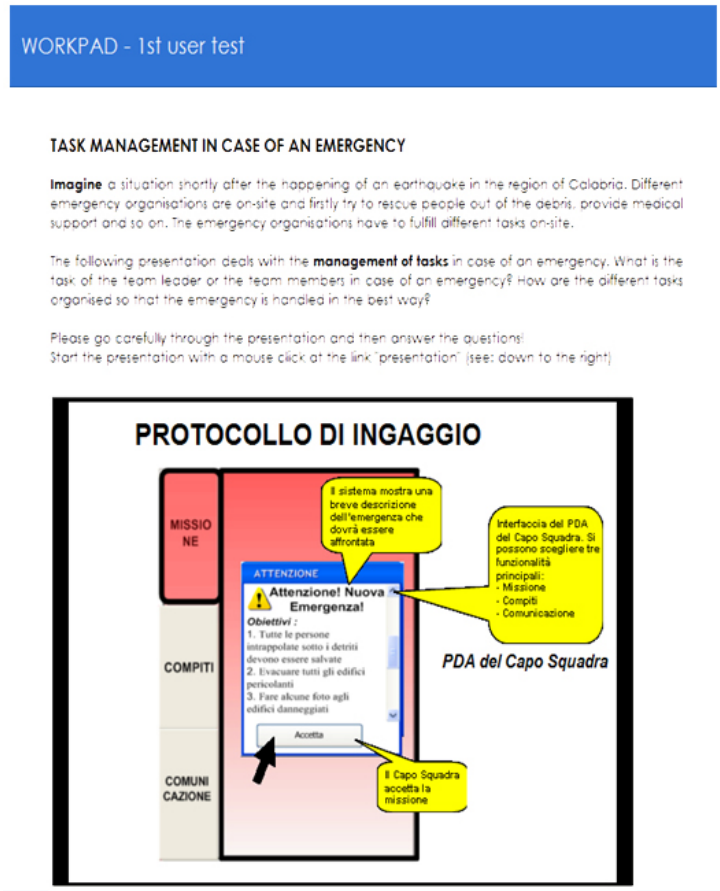


Figure 6.1: A Screenshot of Online Pre-Test - Task Management

nator at regional level, management of integrated projects in crisis management for Civil Protection, or risk prevention.

6.1.2 Conclusions and Recommendations

This subsection summarizes the main comments and feedback from the users as recommendations for improving each component. Please note that these statements are not quotations, but already interpreted recommendations. Generally, Figure 6.3 depicts the level of experience of the test users in using PDAs. 46% were completely inexperienced which is an interesting fact since the selection of types of emergency operators is representative.

Task Management

A screenshot representing the mock-up of the Task Management component is shown in Figure 6.4. To summarize, it can be said that on the whole the task man-

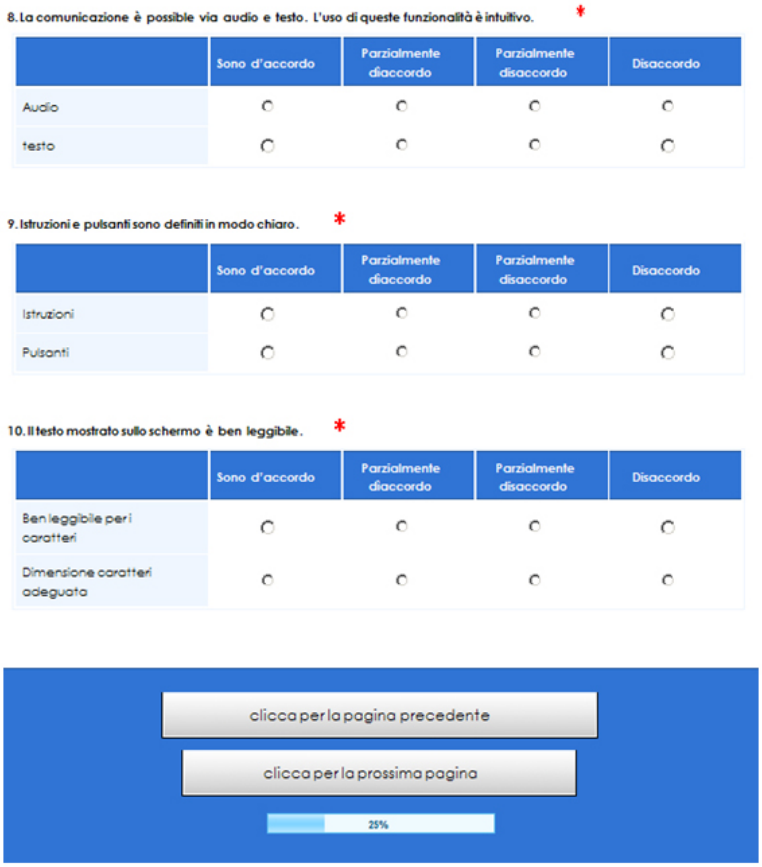


Figure 6.2: A Screenshot of Online Pre-Test - Questions about Task Management

agement component was evaluated positively by the users. The below mentioned comments have been integrated into the component in order to improve the different components and guarantee the satisfaction of the users.

Concerning the improvement of the task management interface, six test users provided comments which also served as recommendations. These are:

- Use textual commands or descriptions besides icons (sometimes icons are not self-explaining).
- Users should not be forced to write down longer texts because in an emergency scenario there is often no time to manually edit text fields. So, predefined inputs (e.g., drop-down lists) are recommended.
- Make less intermediate steps. Users should be able to carry out their intents faster.
- Insert toolbars with frequently used functions represented by an icon.

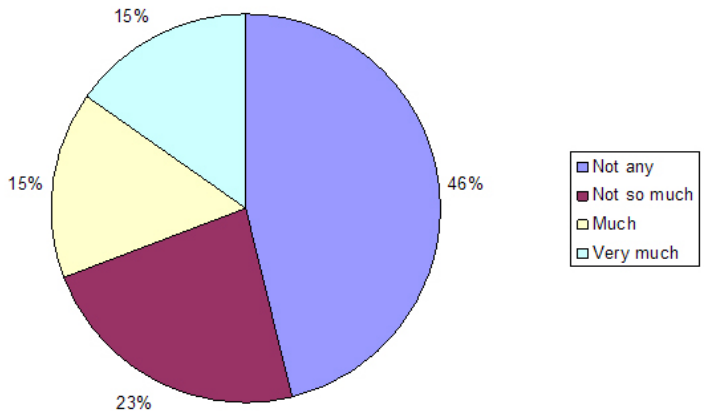


Figure 6.3: Users' Experience with PDAs

- Buttons and main functions should be better explained. This can be supported by simple help guides (e.g., tool-tips).

Three users mentioned general proposals for the improvement of the task management component. These are:

- Increase the number of default (=pre-defined) messages or texts.
- Provide a possibility for planning a coordination meeting among Back-End Coordinators and Front-End team operators in order to have means for discussing the task flow again.
- Make use of compact summarizing windows.

Finally, one user said that it would be good to define collaborative working spaces among instructors, team leaders and team members which make the exchange of additional information and opinions possible. A recommendation which came from another user is to better highlight the priority of tasks.

Map Overview

Concerning the Map Overview component it can be summarized that the users understood the purpose of this component and were of the overall opinion that it is intuitive and understandable in usage. A screenshot representing the mock-up of the Map Overview component is shown in Figure 6.5.

Figure 6.6 gives an overview of possible layers that would make sense according to emergency operators. The x-axis lists the object names and the y-axis the number of testers who voted for the mentioned layers. Figure 6.6 shows that the most important map layers for the test users are: control centres, buildings, information on infrastructure, sensitive objects and public buildings. One test user

Figure 6.4: Screenshot of the Task Management Mock-Up

mentions that it would be interesting to add weather forecasting information which could be used in case of fire emergency in open areas.

Figure 6.7 summarises the answers concerning the frequency of necessary coordination activities between colleagues.

Two open questions concerning functionalities in the questionnaire present proposals for improvement of the map overview functionalities. These comments have been taken into account by the programmers in order to improve this component.

Test users propose the following improvements concerning functionalities:

- Display x and y coordinates in form of latitude and longitude values.
- Insert a warning system based on chromatic color-codes to signal dangerous objects (e.g., a particular dangerous area or building) or cases (e.g., team operators arrived in the critical zone or the presence of people to be saved)
- Increase ID character size of the team members! It is difficult to read in the map.
- Communication between map applications should be possible by pictures and videos.

Test users mentioned further functionalities that should be included in the map overview component in order to support their daily work:

- Show rescue vehicles (ambulances) in the visualized areas.
- Enable zoom in or out in order to make critical points clearer (e.g. in buildings) which may not be evident at a first glance by expert people.
- Make the visualization of shortest alternative paths possible - especially for rescue teams this function could be helpful.

Figure 6.5: Screenshot of the Map Overview Mock-Up

- Provide a key that explains symbol and icons.
- Show coordinates and scales on the map.
- Provide a map printing functionality.
- Include additional information, such as weather forecasts. This information could be used, e.g., in case of fire emergency in open areas (under the hypotheses of incoming rain etc.).

Connection Establishment

A screenshot representing the mock-up of the Connection Establishment component is shown in Figure 6.8.

Ten users answered the questions referring to the connection establishment component. Nobody missed a functionality of this component. One recommendation was to include drop-down menus in order to get the information which team members and which work groups are currently available. Members who are currently not available should not be shown in the drop-down menu. Another recommendation was to pre-configure the PDA according to the different organizations (e.g. Police) in order to get only the relevant information.

The GUI of the connection establishment component per se offers less configuration and interaction functionality than the task management and the map

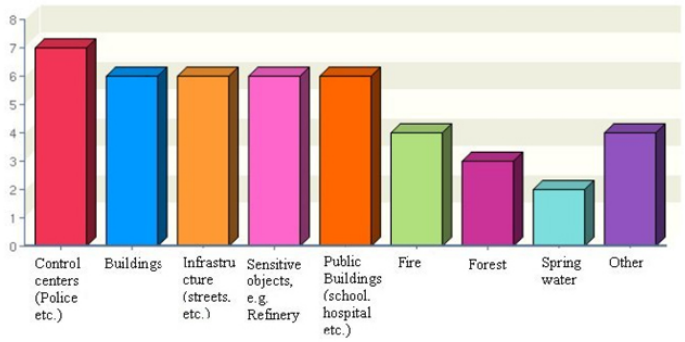


Figure 6.6: Geographic Objects

overview component. Therefore, fewer questions have been composed. The same is true for the Multimedia Editor, the Context Editor and the File Share component which follow below.

Multimedia Editor

A screenshot representing the mock-up of the Multimedia Editor component is shown in Figure 6.9.

Twelve users answered the questions referring to the multimedia editor. This component is comprehensible and intuitive to the users except one user who mentioned that it required knowledge about image editor applications in order to be able to understand this component. Another recommendation was to make it possible to take pictures of the actual emergency situation and share them with the other team members.

Context Editor

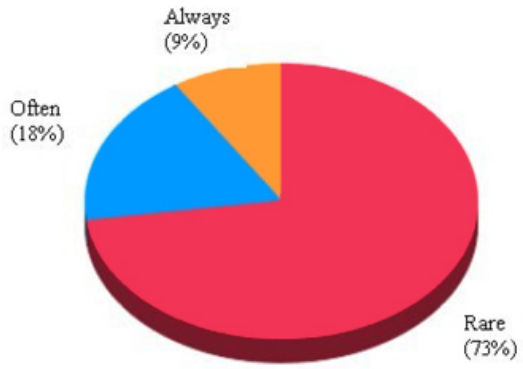
Ten users held the opinion that the context editor interface is intuitive and understandable. Two test users said that it is not clear for understanding and less intuitive. Figure 6.10 illustrates these results.

Comments and recommendations users mentioned were:

- Eliminate some items. Simplify the user interface. Sometimes it seems to be a bit overloaded.
- Insert a help function in order to assist team operators in gathering (emergency-related) information.

File Share

All users were content with the file share component. For ten out of twelve users this component is relevant for their daily work. One user gave the recommendation



Item	Count	Percent %
Rare	8	73%
Often	2	18%
Always	1	9%

Figure 6.7: Coordination Activities with Colleagues

to consider group based file sharing functions. Concrete recommendations were:

- Provide the possibility to send one file to one or more people in one step.
- Provide the possibility to send one file to the whole peer group (=work group).

6.2 Controlled Experiments

The second step of the WORKPAD validation plan also included the performance of Controlled Experiments to be held in a lab environment under controlled conditions.

6.2.1 Execution Details

The execution of the controlled experiments with the different technical components of WORKPAD was conducted in two steps. First each partner in WORKPAD presented its own component, explained the functionalities and several specific details to other partners of WORKPAD. Second, we have set up an additional, bilateral meeting with the user partner Civil Protection Calabria (PCRC) in Reggio Calabria, with the purpose to show them the components.

The users looked carefully through the components, tested them, and provided comments about how to improve the first prototypes of the different WORKPAD components. Due to language issues there was no way around in splitting this test

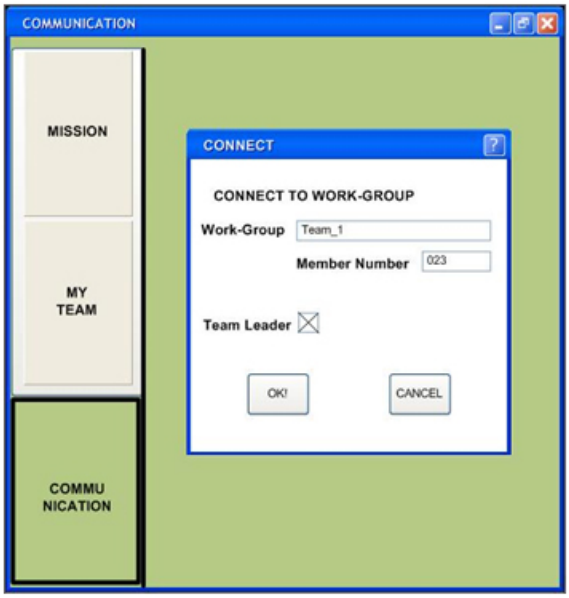


Figure 6.8: Screenshot of the Connection Establishment Mock-Up

type of controlled experiments into two parts. The first technical experiments had to be in English for the developing partners. The second experiments, however, had to be in Italian for the non-English speaking emergency management operators.

6.2.2 Conclusions and Recommendations

PCRC provided hints about how the components have to be improved in order to meet their expectations and mentioned the following recommendations for improving WORKPAD:

- It is very important that the user interface of the WORKPAD system is easily understandable and easily usable as the emergency operators are in a special situation (stress, sometimes dangerous, ...) in an emergency.
- The different components shall be integrated so that it is possible to deal with a single component when using the system.
- The users showed us their currently running GIS on their computer. They gave us hints on which data to use for the GIS client.
- Concerning the Task-list Handler the users mentioned that it will be very helpful and save them time in case of an emergency.

Figure 6.9: Screenshot of the Multimedia Editor Mock-Up

6.3 Cooperative Evaluation

The Cooperative Evaluation is one further important step in order to ensure a "usable" interface according to the User-Centered Design approach followed in WORKPAD. The results are respected and included in the further development and integration activities and therefore contribute to the improvement of the system.

6.3.1 Execution Details

This user test intended to reach the following goals:

- Test WORKPAD with the first "real" prototypes.
- Estimate the performance of specific tasks in order to gather information on usability issues of the system.
- Derive results of the user test and provide recommendations to the developer team in order to improve WORKPAD.

PCRC organized 20 people who were willing to perform the user test and get to know more about the WORKPAD project. All 20 test users were introduced into the WORKPAD project, 4 users had the opportunity to intensively test and gave

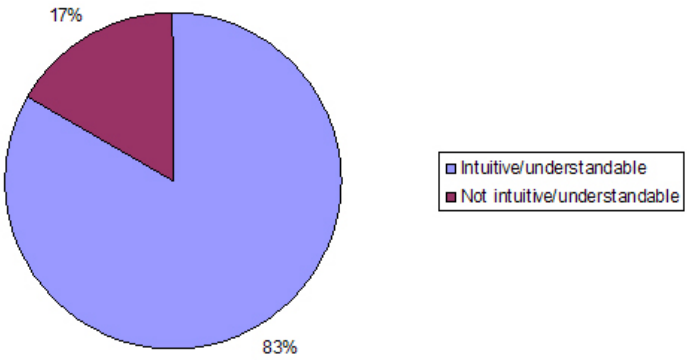


Figure 6.10: Intuitiveness of the Context Editor

comments and recommendations concerning the improvement of WORKPAD. All four male users came from the PCRC organization "Le Pantere Verdi" and were below 30 years old. They were responsible for carrying out different tasks in PCRC. The other 16 users watched the test performance and also provided comments after finishing the test. Each user test took about 1,5 hours of time. We performed a cooperative evaluation with the first real prototypes in two parallel sessions with one user each. The users were asked to perform several tasks with the different WORKPAD components.

The user tests were audio/video recorded and were organized in the following way:

- Introduction of the whole WORKPAD system to all present users (20 potential users from PCRC) and explanation of the purpose of the user test.
- Formation of the first two parallel sessions.
- The moderator explained the first WORKPAD component to the user.
- Performance of the first task through the user.
- The moderator explained the second component to the user.
- Performance of the second task through the user and so on.
- Performance of a video analysis in parallel in order to gather impressions that the user had from the software; the user test was also audio taped.
- Presentation of the video to the user - the moderator wrote down the comments of the user.
- Performance of a semi-structured interview with a questionnaire.

6.3.2 Conclusions and Recommendations

This part deals with the conclusions and recommendations that the users mentioned with regard to the whole WORKPAD system and also the different components. Most of these recommendations were actively integrated and improved by the WORKPAD partners in the delivery of the further software. Some recommendations, however, were out of scope of the project or not realistic. Figure 6.11 gives a summary about the main statements collected through this user test. These statements cover the usefulness, the stability, and the attractiveness of the prototypical WORKPAD components.

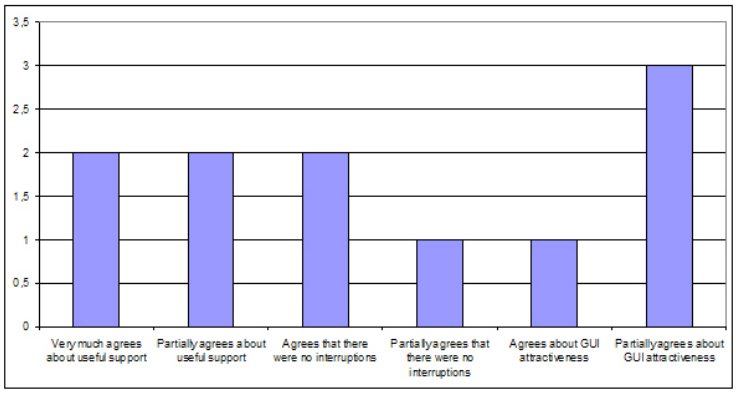


Figure 6.11: Summary of the Cooperative Evaluation Results

General comments relevant for the overall system behaviour

- The information exchange in WORKPAD has to be done fast because if this is not the case the information becomes "out-dated" and therefore is no more relevant to the users.

- The colours which are used in WORKPAD should be significant (e.g.: red for emergency, yellow for fire, etc.).

- It would be useful to make an integration with other instruments (from aeroplanes, boats, etc.) possible.

  - WORKPAD will not be interoperable with other instruments like aeroplanes. This is a good comment, but it is not in the focus of the project.

- The battery life of the PDA could be a bottle neck, e.g., if the operator leaves in the morning and comes back in the evening.

Comments on the Task-list Handler component

- The connection should be faster (e.g.: it takes time to download the right mission and then connect).
- It is very easy and simple to use.
- At the time of the test the system sometimes crashes. This should be improved.
- It would be good to enable voice communication.
- The user is interested to get to know how to add tasks (mission set up).
- A very general comment concerning the Task-list Handler was to handle more and more complex information.
- The Task-list Handler helps to organise tasks and operators without confusion.
- The user very much appreciates that he does not receive the next task until he has finished the one which is currently running.

Comments on the GIS Client component

- The GIS Client is very easy to use. It looks like a Google Maps application.
- The application runs without difficulties.
- The included map icons should be a little bigger.
- One user especially appreciates the position monitoring and the real-time information exchange.

Comments on the Fileshare component

- The Lightweight storage is a very helpful component if it worked (one user test did not work, the application crashed).
- It is very good to be able to exchange photos in real-time.
- The interface of the lightweight storage component shall be improved.
- The Lightweight storage component is good, useful and easy to use.
- Proposal for improving the usage of the component: it should be possible to directly open the photos instead to manually access the PDA file system.

Comments on the Context Editor

- It is very easy to send data.
- The Context Editor is usable.
- The Context Editor works without difficulties.
- The GPS position should be integrated, instead to manually fill in the position (latitude and longitude); at least the current position shall be suggested.
- The information which can be filled in the Context Editor is detailed enough.
- The Context Editor should be improved with more icons in order to make it clearly arranged.

6.4 Tests with External Users

After the performance of the user test with expert users, we accomplished another usability test with non-expert users (people outside of the domain of emergency management). The intention of this test was to get also feedback on the user interface from persons who are not familiar with the emergency topic.

6.4.1 Execution Details

Each project partner with a Front-End component tested the graphical user interface of his component with 4 to 6 people. In this test very experienced users, but also users with no experience participated. 7 users were "not so much" and 2 users were "much" experienced (see Figure 6.12). On the whole 21 people performed the test.

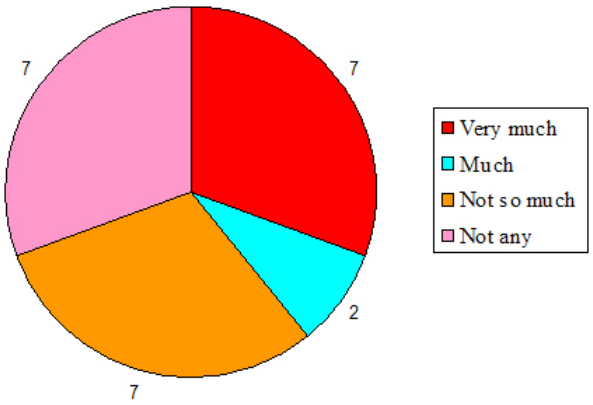


Figure 6.12: Experience in working with a PDA

The test was executed taking into account that each project partner with a Front-End component was responsible for testing his component with 4 to 6 users. Fist we explained the WORKPAD project, the intention of the usability test to the users and we also presented the main functionalities of the components. Then each user went through different tasks with the component and responded to questions concerning the usability of the component afterwards. In the following subsection 6.4.2 the test results are presented based on the results of the questionnaires (listing of all questions and the additional answers of the users) and analysed with regard to the different WORKPAD components.

6.4.2 Conclusions and Recommendations

In this subsection we provide a summary and graphs of the main test results with regard to the different WORKPAD components that were tested.

GIS Client

Six external users evaluated the GIS Client - three male and three female users. Especially concerning the intuitiveness of the component the users were not fully satisfied with the GIS Client. They mentioned for example that the terminology should be more intuitive and that the component is too slow for the tasks in case of an emergency that have to be performed. The users did not have any application crashes and very much agreed that the screen design is easy readable and also understandable. Figure 6.13 summarises the statements regarding the attractiveness of the screen design.

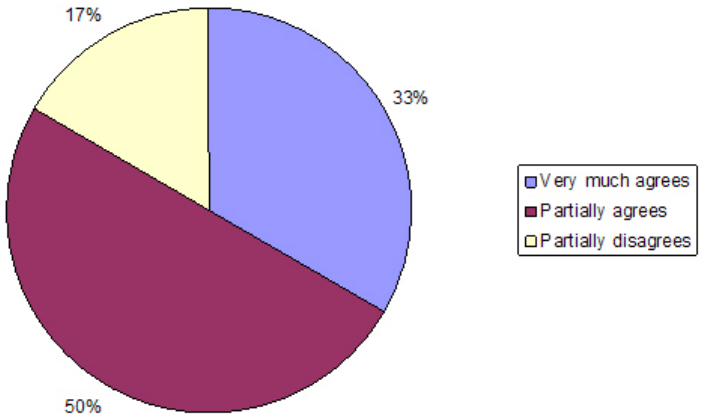


Figure 6.13: GIS Client: Attractiveness of the Screen Design

1. **The GIS client is intuitive and easy to use.** One half of the users stated that the GIS client is easy and the other half said that it is not easy to use. Comments and recommendations of the users:

- The map interaction is too slow compared to currently existing map applications from Google Maps etc.
- Display of the PDA it too small, because it is difficult to read the signatures.
- The layout and usability of the PDA is outdated; usage of touch interaction would be good and up to date.
- The usage of the PDA is less intuitive, for example the sensibility of the context menus is not well adjusted.
- Use "exit" instead of "quit", because everybody is more familiar with that term in this sense.
- One user said that for example the "Area Dialogue" closes if you want to edit something.
- The used terminology should be more intuitive.

2. **The GIS client runs without interruptions and without crash.** Three users did not have any problems with crashes and interruptions, one said that the component ran well and two stated that the GIS client ran not so well.

3. **The screen design of the GIS client is attractive.** Two users very much agree that the screen design is attractive, three partially agree and one user disagrees. Comments and recommendations of the users:

   - The screen design of the component could be improved, but nevertheless it is better to focus more on the provision of functionalities than too much on the interface design.
   - The "exit" functionality has to be available in every modus e.g. also in the layer menu.
   - Use the term "cancel" instead of "back" and put it on the right on the screen and put the "OK" button on the left.
   - Provide the "multi-select" functionality for selection boxes.
   - Use a better option to click off a menu

4. **The screen design is consistent** (for example: meaning of icons and screen layout, placement of icons, map symbols etc.). Five users very much agree that the screen design is consist. One user stated that she could not give a judgement as there is only one "real" view.

5. **The screen text is easy readable.**

   (a) Easy to read font types

   (b) Adequate font size

   (c) Text layout supports readability

Three users very much agreed on the statement "the screen design is easy readable" and three partially agreed. Comments and recommendations of the users:

- Menus should be bigger in size.
- Improve: Labels are overlapping.
- Provide better structures concerning the layers
- It should be possible decide as a user which labels shall be used and which not.
- The labels on the map are coloured white in the background, but the white colour is not always in the background of the whole lettering.

6. **The GIS client is easy to navigate.** Two users very much agree that the GIS client is easy to navigate, three partially agree and one user partially disagrees. Comments and recommendations of the users:

   - "Exit" should be renamed in "cancel", because this is confusing.
   - Currently the performance of the application is rather slow.
   - For example: functionality "create annotation" - multiple choice and combo boxes shall be available, at minimum one name and one status for input should be provided.

7. **What are your recommendations for improving the graphical user interface of the GIS client?** Comments and recommendations of the users:

   - Use a variation of line thickness at different zoom labels.
   - Take a look at the problem of overlapping labels.
   - Hierarchies are mixed up within the different layers.
   - Problems with labels: it should be possible to blind them out.
   - Use less details and provide more usage "on-demand".
   - Use "better" and more understandable icons.
   - It would be good to include a buffer functionality and to be able to specify a query.

8. **Are the different steps that you had to perform in order to finish a task comprehensible for you?** This question was of no relevance for the analysis, but it gave us a scope for the next user test and the tasks that had to be designed.

Filesharing component

Five external users evaluated the Fileshare storage component. One female user participated in the test. The users had problems with the navigation within the Filesharing component. Concerning the other usability issues (crashes, intuitiveness, etc.) the users were enough satisfied.

1. **The Fileshare component is intuitive and easy to use.** Three users said that the Lightweight storage component is easy to use and one user mentioned that it is not so easy.

2. **The File Share component runs without interruptions and without crash.** Four external users stated that the component runs good and one said that it runs not so good.

3. **The screen design of the Fileshare component is attractive.** Three users partially agree that the screen design of the component is attractive and one user partially disagrees.

4. **The screen design is consistent** (for example: meaning of icons and screen layout, placement of icons, map symbols etc.). One user very much agrees, three users partially agree and one user partially disagrees that the screen design is consistent.

5. **The screen text is easy readable.**

   (a) Easy to read font types

   (b) Adequate font size

   (c) Text layout supports readability

   Four users partially agree and one user partially disagrees that the screen text is easy readable.

6. **The Fileshare component is easy to navigate.** Two users very much agree and three users partially disagree that the Fileshare component is easy to navigate.

7. **What are your recommendations for improving the graphical user interface of the Fileshare component?** One user stated as recommendation that a touch screen instead of pencil entry should be available.

Back-End configuration application

Four male external users evaluated the Back-End query application. The Back-End query application needs a simpler user interface and a better navigation in order to meet the expectations of the users.

1. **The Back-End query application is intuitive and easy to use.** Three users said that the application is intuitive and easy to use and one user stated that it was not so easy and intuitive in usage. Comments and recommendations of the users:

   - It is intuitive and easy when simple tasks are performed. Rather, when you need to specify more complex things such as restrictions is a little bit more complex.

2. **The Back-End query application runs without interruptions and without crash.** Three users said that the application runs very good and one user had problems with running the application.

3. **The screen design of the Back-End query application is attractive.** Two users very much agreed and two users partially agreed that the screen design of the Back-End query application was attractive. Comments and recommendations of the users:

   - There are too many options available, so it is difficult to find what is needed.
   - It could improve to some degree. User experience is better if the component is used with some plug-ins which improves visualisation.
   - Colours should be personalizable.

4. **The screen design is consistent** (for example: meaning of icons and screen layout, placement of icons, map symbols etc.). Two users very much agreed and two users partially agreed that the screen design of the Back-End query application was consistent. Comments and recommendations of the users:

   - The user interface is consistent, but some practice is needed before it can be used efficiently.
   - Some icons should be better comprehensible.

5. **The screen text is easy readable.**

   (a) Easy to read font types

   (b) Adequate font size

   (c) Text layout supports readability

   Four users partially agreed and one user partially disagreed that the screen text was easy readable.

6. **The screen design of the Back-End query is easy to navigate.** All four users partially agree that the application was easy to navigate. Comments and recommendations of the users:

- It should be permitted to define properties directly from the classes.
- The way you navigate is standard.
- If you close a tab accidentally, you get in trouble as there is no way to reopen it.
- Sometimes you get confused, because available options are not fully explained.

7. **What are your recommendations for improving the graphical user interface of the Back-End query application?** Comments and recommendations of the users:

   - It would be useful to allow editing on the graphical layout and to provide more integrated drag and drop actions.
   - A GUI assistant that suggests some steps in creating an ontology would be useful for non-expert users.
   - The user interface should be simpler for a non expert-user. A contextual guide would be useful.
   - One user recommends to open a new tab or panel only if requested and not by default.
   - It should be permitted to write directly in OWL (for expert users).

Task-list Handler, Context Editor and Multimedia Editor

Three male and three female users performed the usability test. With regard to these components the users stated that especially the navigation should be improved so that it is clearer and things can be found more easily. Referring to the attractiveness of the component, three users were satisfied and three users were not so satisfied.

1. **The Task-list Handler, Context Editor and Multimedia Editor are intuitive and easy to use.** Five users said that the application was intuitive and easy to use and one user stated that it was not so easy and intuitive in usage. Comments and recommendations of the users:

   - The movement and navigation in the applications is not very intuitive. I miss guidance of the user.
   - The Task-list Handler and Context Editor are easy to use, but the Multimedia Editor is quite difficult in usage.
   - Sometimes there are long response times and I was not sure of "progress" - the application seems inactive.

2. **The three applications run without interruptions and without crash.** Four users stated that the applications run good, one user mentioned very good and one user said not so well.

3. **The screen design of the three applications is attractive.** Three users very much agreed and three users partially disagreed that the screen design of WORKPAD was attractive. Comments and recommendations of the users:

   - Design is not intuitive, buttons are missing (refers to changing of start and stop button).
   - In wide mode the main window seems less readable.
   - The design is alright. Some controls have different proportion (buttons are too large, some text is inefficiently small, but still readable).
   - Readable font, good layout of controls.

4. **The screen design is consistent** (for example: meaning of icons and screen layout, placement of icons, etc.). Three users partially agreed, two users very much agreed and one user partially disagreed that the screen design was consistent. Comments and recommendations of the users:

   - Multimedia editor is less intuitive.
   - It is important to keep some places for controls.
   - The icons are OK.
   - It is not good that buttons disappear after you press them.

5. **The screen text is easy readable.**

   (a) Easy to read font types

   (b) Adequate font size

   (c) Text layout supports readability

   Four users very much agreed that the screen design was easy readable, one user partially agreed and one user partially disagreed that the screen text was easy readable. Comments and recommendations of the users:

   - The text font could be larger.
   - The text is readable, but this depends very much on the lighting conditions.

6. **The screen design of the three applications is easy to navigate.** Four users very much agreed, one user partially agreed and one user partially disagreed that the three applications were easy to navigate. Comments and recommendations of the users:

   - Because of the long response time and the missing bar e.g. "work in progress" the navigation is not much transparent.
   - Guidance of the user is missing e.g. when you make mistake there is no way back ? closing the Context Editor before completion of the form and then the Context Editor can not be started again.

- Applications shows appropriate actions in good actions.
- The style is OK, but I am not sure how it is when firefighters are in action.

7. **What are your recommendations for improving the graphical user interface of the three applications?** Comments and recommendations of the users:

   - If some capabilities are set before the mission, they should be different from others.
   - Better placement of icons and of the main window.
   - The font should be larger and the contrast to the text should be better.
   - Recommended time of completion of a task should be displayed.
   - In the Multimedia and Context Editor the button to come back to Task-list Handler should be named "back" instead of "quit".
   - Provision of the option to step back.

6.5 ShowCase - Without WORKPAD System

This first scenario was performed in a realistic setting with the purpose to simulate a real emergency with emergency operators. Particular storyboards and selected tasks out of the emergency scenario were showcased. In this first drill operators acted with their current knowledge and technologies, without the support of the WORKPAD system. It was very helpful to watch and document the work of the different emergency organizations during the response phase of an emergency. This showcase was intended as a proof of concept of the design of the various storyboards for the real showcase with the WORKPAD system.

6.6 ShowCase - With WORKPAD System

This section summarises the final showcase of the WORKPAD project, held in Pentidattilo (a Calabrian City). This was the last and final type of user test we performed. This Section is subdivided into further subsections describing execution details, which cover the involved user organisations and technical details, and the conclusions and recommendations.

6.6.1 Execution Details

Here, we provide a summary of the showcase that was executed to show and evaluate the prototypical implementation of the WORKPAD reference architecture for collaborative work. We decided to execute three storyboards as this implies the

generation of more test results that we can analyze and discuss. All these storyboards cover phases of an ongoing emergency. Additionally, some days before the showcase starting, we decided to add one extra storyboard that shows the information preparation and semantic integration of the Back-End systems of the organizations during peace times. Hence, at the end we executed four storyboards during the showcase days in Pentidattilo.

Before to effectively started the showcase, we introduced the WORKPAD idea and system to the users who took part in the showcase. After that general introduction, we provided the possibility to actually use the software hands on and we provided some simple training such that the users could get a better idea.

The four storyboards were conducted all in one same day. We started with the storyboards 1, 2 and 3. For each storyboard we performed a trial run already with the users in order to check the functionality in a final test. Technicians were available for questions that might arise from the users. After that, the real execution started. In that case, technicians were available for assistance too but only in very urgent cases.

Moreover, we decided to run the fourth storyboard about the BE data integration in peace times indoor. Storyboard 4 was designed slightly different from the others. We did not execute a drill with real users and measured the behavior. In that case and as the topic of semantic data integration is rather complex, we decided to give a demo and provide in-depth explanations and hands on opportunities to the users.

In order to have documented results, every operator was followed during the drills by people from the project consortium. These people took notes on Task Execution Forms (see Figure 6.14) that we specifically designed for recording relevant data and metrics. After the execution each operator was available for interviews that we conducted in a structured way according to a pre-defined questionnaire forms (not shown here for space reasons). All the storyboard executions were additionally recorded on tape such that finally a better impression of the drills can be depicted.

6.6.2 Summary of the Storyboards

This section summarizes the main ideas about the four storyboards.

Storyboard 1 : Documenting an Area

Storyboard 1 demonstrates various tasks that are necessary during initial phases of an emergency where operators need to get an overview about the situation on site. Different operators collaboratively collect information with various tools. The following three figures describe this process in terms of an UML activity diagram. Figure 6.15 illustrates the main overall process of SB1, while Figure 6.16 and Figure 6.17 give details about the two contained sub-processes "Assessing Buildings" and "Investigation of People".

STORYBOARD

Name Firefighter:	
Name Companion:	
Task name:	

1) Time stamp

Time stamp task begin:	
Time stamp task end:	

2) Number of required assists: _____

3) Correct task outcome

Yes	
No	
If no, please write down the reasons	

4) Number of errors:

Figure 6.14: The Task Execution Form

Storyboard 2 : Establishing a Medical Point

Storyboard 2 subsumes the activities that are necessary for planning and creating a medical point for first aid at an emergency site. This storyboard includes tasks such as finding ideal locations, building up medical tents and furnishing the tents with medical equipment. Figure 6.18 describes the main process, where Figure 6.19 denotes details about the contained sub-process about building one medical tent.

Storyboard 3 : Evacuation of people

Storyboard 3 deals with checking the presence of victims on an emergency site, collecting critical information on-site and distributing this information among the team members. As a first activity each team member should get an overview about the current position of the other team members. After that different relevant tasks about checking the situation on site are assigned to individual team members. Figure 6.20 gives an overview of that process encapsulated in storyboard 3.

Storyboard 4 : Configuration of the Back-End Data Integration

As argued in the subsection 6.6.1, we decided to add one extra storyboard that shows the information preparation and semantic integration of the Back-End systems of the organizations during preparation times. Hence, storyboard 4 shows how organizations should configure their data sources like, for instance, geologi-

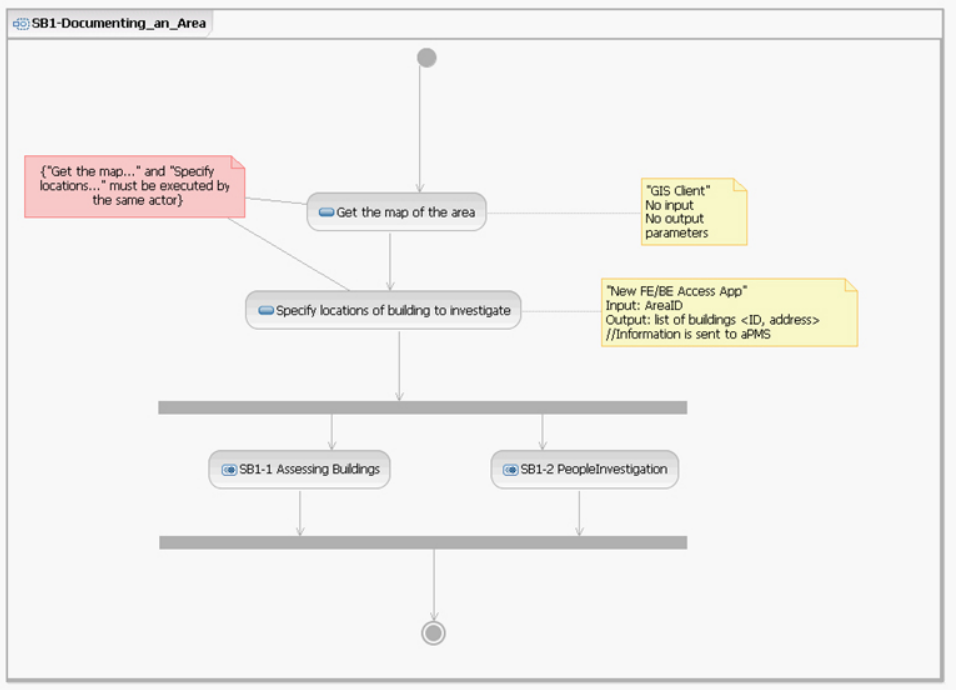


Figure 6.15: Storyboard 1 : Main Process

cal or demographic data. Then these data can be combined with data from other organizations. Figure 6.21 shows a screenshot of the configuration software for the Back-End data integration subsystem of WORKPAD that was presented in the scope of storyboard 4.

6.6.3 Conclusions and Recommendations

For each task in each storyboard the following indicators were collected by WORKPAD personnel during the showcase execution (see task execution form in Figure 6.14) :

- Time span.
- Number of required assists.
- Correct task outcome.
- Number of errors.

Each storyboard was performed twice. First, we conducted a trial execution and then the "real" showcase was executed. Our analysis also integrated the trial execution. With this information we could draw conclusions whether emergency

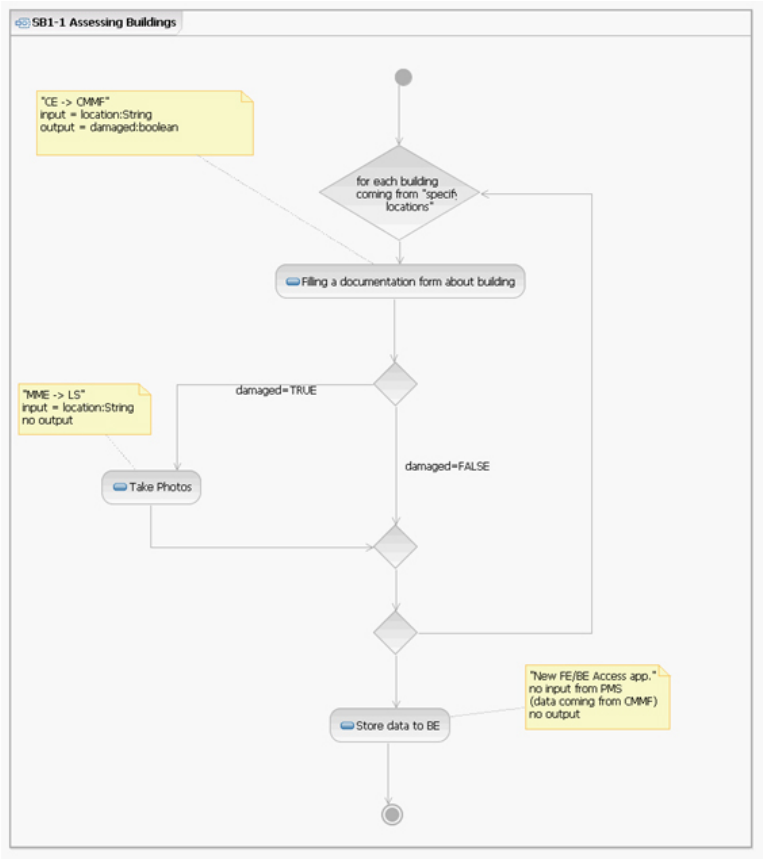


Figure 6.16: Storyboard 1 : Assessing Buildings

operators had problems with using the system or not and how long it took the user to execute a task. WORKPAD project members accompanied the emergency operators and filled in the task execution forms which measured the above mentioned indicators. As soon as the users finished the execution of a storyboard we interviewed them in order to get information on user satisfaction.

As the WORKPAD system is based on an adaptive process management meaning that tasks may be distributed differently in different runs according to context, the trial execution and the "real" execution of the storyboards were not identical and hence cannot be correctly compared in a quantitative way. The users more often asked for assistance and also system errors happened more frequently. This means that the efficiency and also the effectiveness were higher in the "real" run than in the trial. According to the correct task outcome it can be mentioned that in the "real" execution the task outcome was always correct.

Figure 6.22 illustrates a summary of the evaluation of the task execution forms. The figure shows the mean values of the defined indicators per task and per storyboard. One interesting fact is that all the means of all values of all indicators

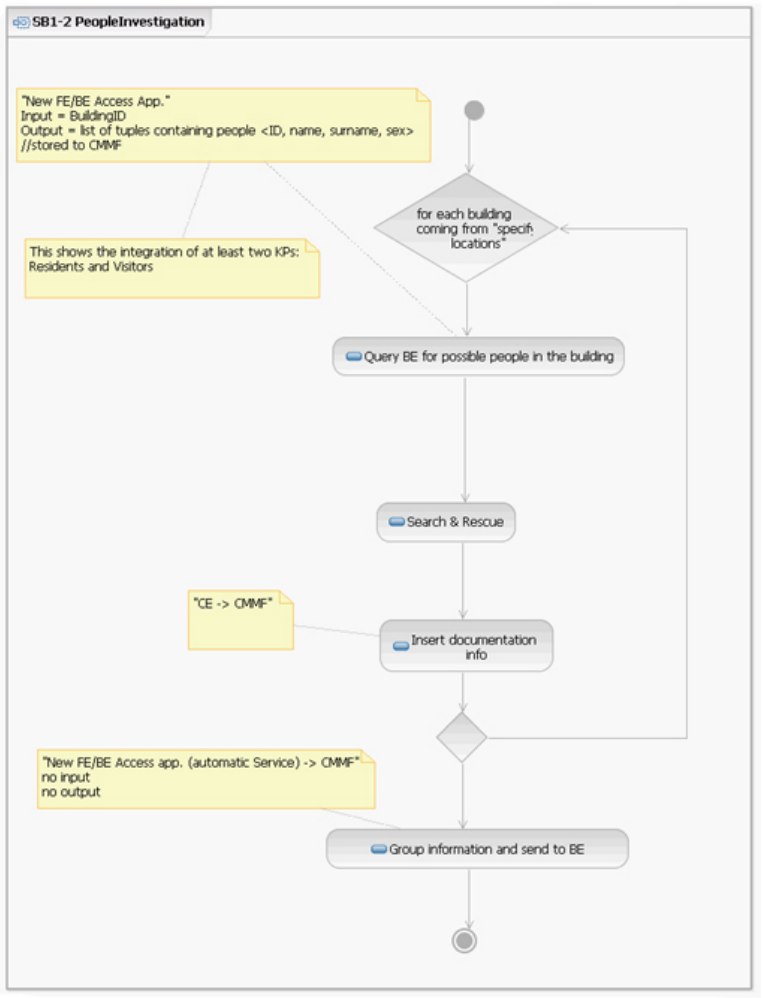


Figure 6.17: Storyboard 1 : People Investigation

improved in the real execution compared to the trial execution. From this result we can conclude that the users got quickly acquainted with the WORKPAD software so that they were able to use it beneficially already after one run. We base this on the User-Centered design approach and the steady feedback from the close collaboration with expert users from PCRC throughout the whole system design and engineering process. The various methodological approaches - as presented in Section 4.3.2 - give evidence about this continuous improvement process. A good example is the indicator "number of required assists, mean per task", which dropped from 2,0 in the trial run to 0,7 in the real run. There is only one exception where the values increased which is the indicator "time span per storyboard" in minutes. The values increased from 24,4m to 35,3m. This can be explained by the fact that in the trial showcase not all the tasks were really executed but only

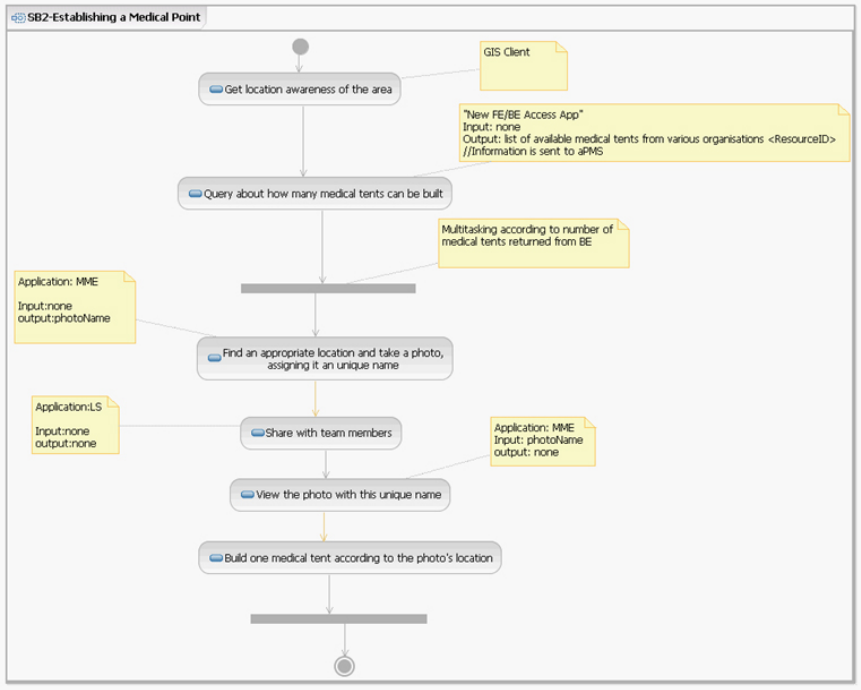


Figure 6.18: Storyboard 2 : Establishing a Medical Point

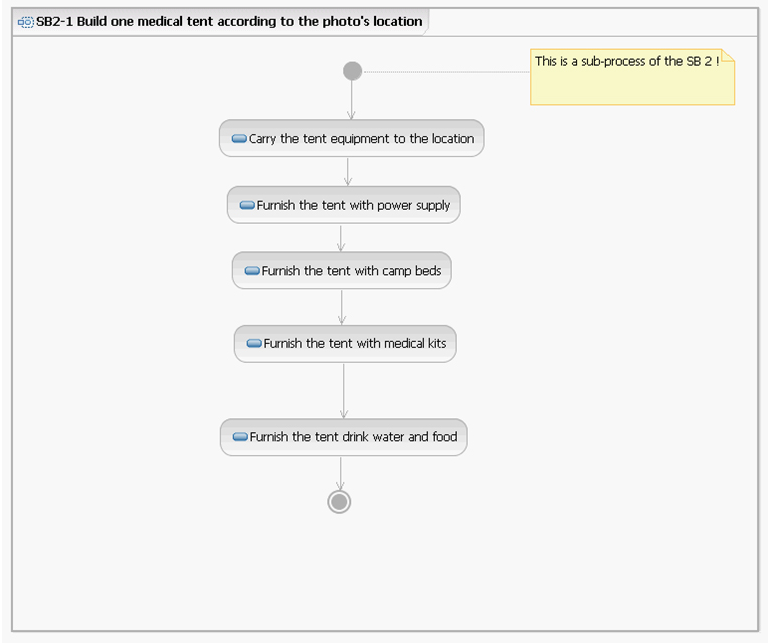


Figure 6.19: Storyboard 2 : Building a tent according to the photo location

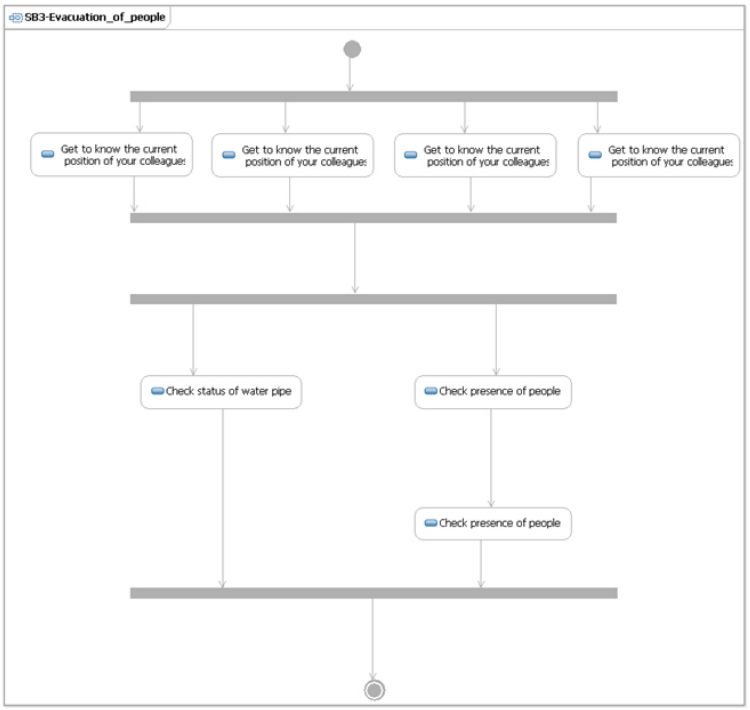


Figure 6.20: Storyboard 3 : Main Process

simulated. For instance, we did not ask the operators to build and furnish the tents in the trial showcase. Of course, in the real showcase the operators built up and furnished the tents, hence the longer execution times. But we did not want to ask the operators to build, remove, and re-build the tents, which would have meant a lot of effort to them.

In order to get information about user satisfaction we performed questionnaire based interviews. We made clear that the system wherewith the users were interacting with was a research prototype and, hence, some aspects might not be perfect. We got the impression that the users very well understood this intention, which was also confirmed by the very constructive feedback. Half of the test users who agreed to make the interview did not have any experience in working with a handheld device. It was the first time for them to work with a PDA. Four of them fully agreed and eight users partially agreed that WORKPAD is intuitive and easy to use (see Figure 6.23). No one disagreed either partially or fully. Some of them had problems with the visibility on the screen because of the sun and others mentioned that the system could react quicker.

Concerning the efficiency of supporting the users in performing their tasks, no user disagreed, six users partially agreed and six users very much agreed that WORKPAD supported them efficiently in performing their work (see Figure 6.24). Some users mentioned issues about slowness, or raised concerns that WORKPAD

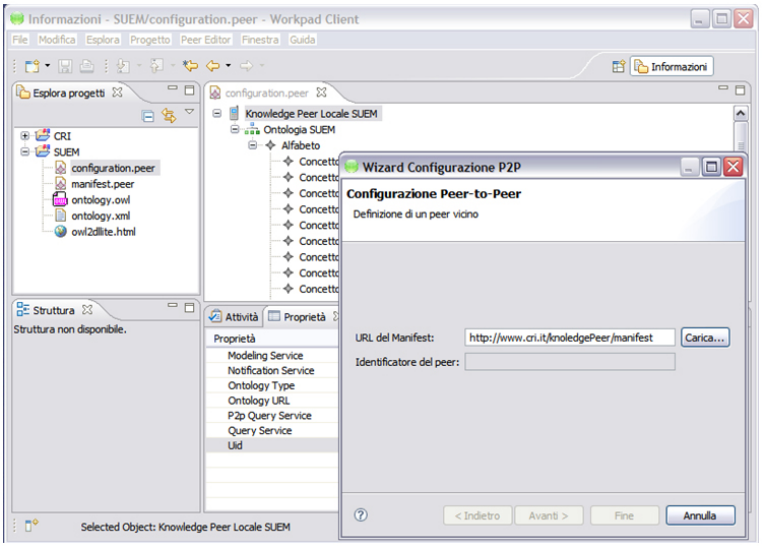


Figure 6.21: Storyboard 4 : Screenshot of the Configuration Wizard

as only one tool would suffice.

One half of the users said that WORKPAD met their expectations and the other half not. Some of the operators wished a faster system including more functions. One user made the proposal that it would be better to use WORKPAD in the medium-term phase.

Around two thirds of the users said that they could imagine using WORKPAD in their daily work when it is improved before and in case of Soccorso Alpino when it works in the mountains (see Figure 6.25).

The biggest part of the users did not have any problems with navigation and mentioned that the system is logical and intuitive to them (see Figure 6.26).

The users considered the following aspects as very useful: information sharing, geographic information and the assignment of tasks. Two thirds of the users mentioned that there may be other useful functionalities without giving more details about these. In relation to this, the communication issue (i.e., to be able to communicate with other team members - by voice in particular) was cited quite often (see Figure 6.27).

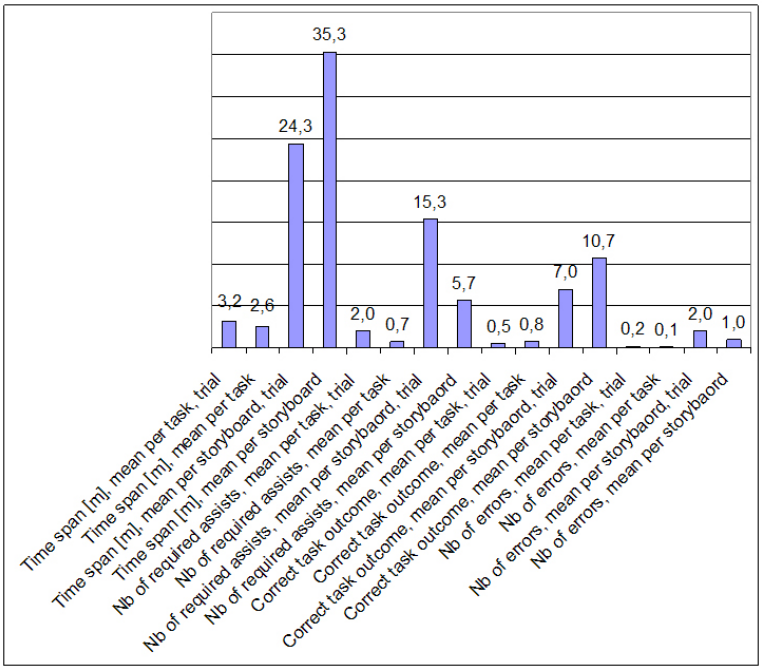


Figure 6.22: Storyboard 4 : Summary of Task Execution Form Results

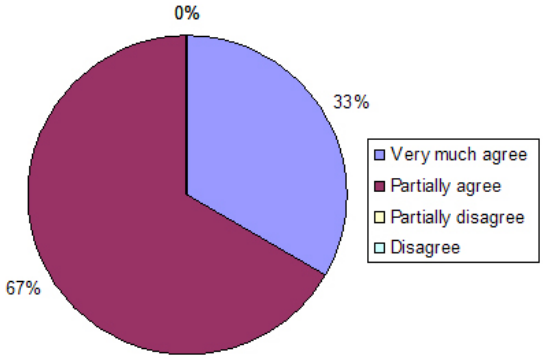


Figure 6.23: Questionnaire: "WORKPAD is intuitive and easy to use?"

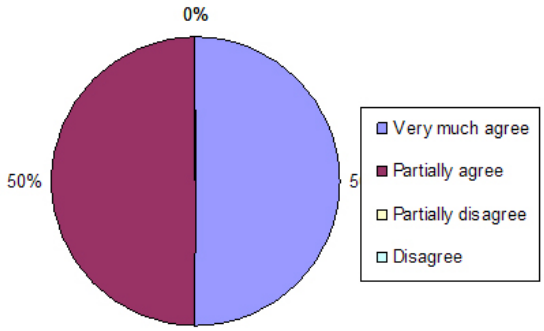


Figure 6.24: Questionnaire: "WORKPAD supported you efficiently in performing the tasks?"

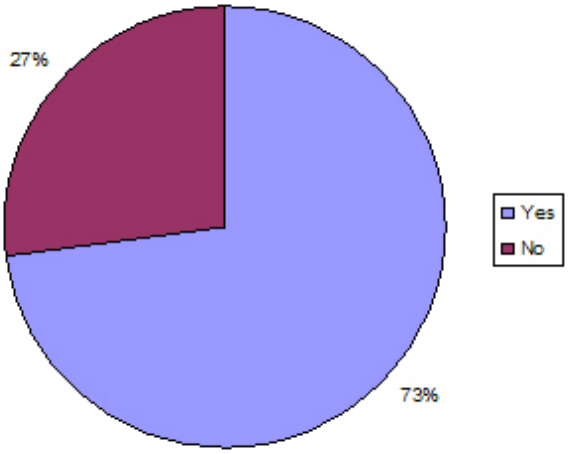


Figure 6.25: Questionnaire: "Can you imagine to use WORKPAD in your daily work?"

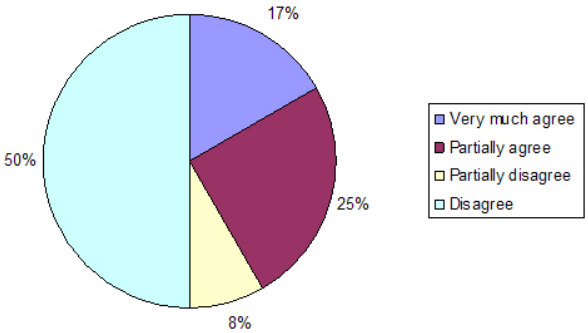


Figure 6.26: Questionnaire: "It is difficult for you to navigate in WORKPAD?"

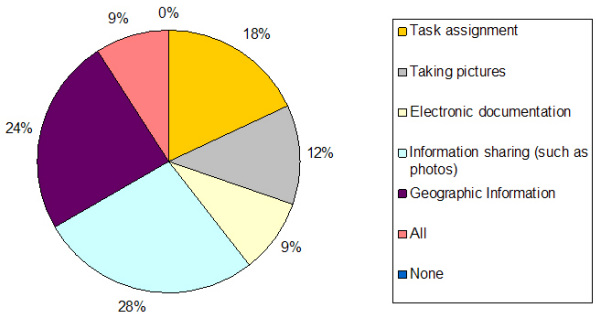


Figure 6.27: Questionnaire: "Which aspects do you consider as very useful?"

Chapter 7

Contribution to the Design and Implementation of the PMS

The teams working in disaster or emergency situations usually use smart devices in very dynamic and mobile scenarios over a network partially unreliable. Activities in critical and emergency scenarios are highly-stressing situations for the users, who generally give more priority on the physical stimuli concerning the activities to execute than on those coming from software applications. Therefore, some challenging issues emerge, which we divided in two categories. The first category concerns grasping the users' mental attentions onto the system as little as possible because pervasive processes are really challenging and stressing for them. The latter category of issues is merely technological and deals with reducing the resource consumptions.

The WORKPAD project, at the Front-End side, has faced these issues by providing (see also Figure 7.1) :

- A Graphical User Interface (GUI), able to organize all the needed information to face an emergency on small screens, such as the PDA ones. When designing client interfaces for mobile, pervasive and critical scenarios, it is important that task-handler interface should attract the user attention only when it is strictly required [50].

- A Process Management System (PMS), able to support operators during Emergency Management for coordinating their activities. In such challenging situations, processes should be adapted in order to cope with anomalous situations, including connection anomalies and task faults. This requires the provision of intelligent support for the planning and enactment of complex processes, that allows to capture the knowledge about the dynamic context of a process [38].

In this Chapter, we show how this knowledge, together with information about the capabilities of the available actors, may be specified and used not only to support the selection of an appropriate set of agents to fill the roles in a given task, but

also to handle the problem of adaptivity. Furthermore, an effective User-Centered Interface developed to interact with the PMS will be show.

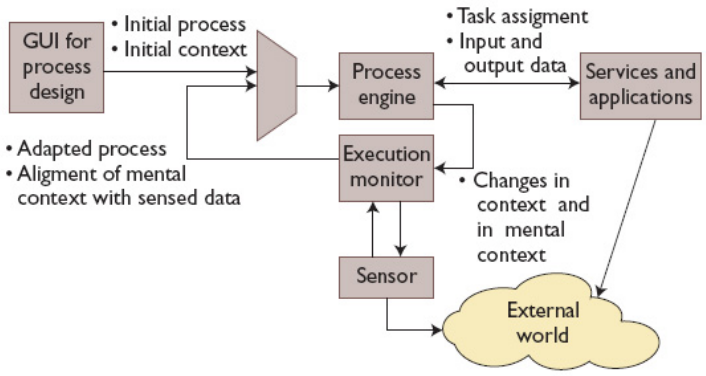


Figure 7.1: WORKPAD's adaptive Process Management System

The Chapter is organized as follow :

- Section 7.1 presents the user interface mock-up and functionalities together with a set of guidelines used for designing it that take into account psychological aspects of the user.

- Section 7.2 describes the engine of the PMS based on well-known Artificial Intelligence Techniques and how it can be extended to tackle adaptation.

7.1 The User Interface

When designing a graphical user interface, the first aspect to consider is how and where the users focus their mental attention. The book [51] defines attention as the totality of information cognitively manipulated by a person. The attention allows also for using human mental resources in a judicious way, by allocating them only on the most interesting inputs. This judiciousness is used to increase the probability of a rapid and accurate answer, giving more priority to the most important stimuli. Indeed, among the great quantity of information made available by senses, memories and thoughts, we can actively manipulate through the attention just a little bit. On the contrary, the information available for the cognitive elaboration, but currently out of our conscious awareness, is found to the level of pre-conscious elaboration. For example, this is why driving a car can be dangerous while talking on the cell phone. The active part that our attention considers in these cases do include driving the car just in a small portion. It is almost out of our consciousness. The greatest part of our cognitive resources is used for the phone conversation, where we are conscious of the words and the sentences used in the discussion with the interlocutor. In this case, these processes are called controlled processes.

The Emergency Management activities represent high-stress situations for the involved actors, which have the necessity to focus their conscious attention, and their efforts, toward the tasks they must complete, ranging from a simple photo taking to the rescue of people under the debris of a building. Therefore, we have designed the interface of our system with the goal of capturing the attention of the actor only when necessary, "stealing" the smaller quantity of cognitive resources.

7.1.1 On Designing User Interface

As far as the User Interface, the WORKPAD's methodology, described in Chapter 4, states that rescue operators must pay attention to the system only when it is strictly needed and, otherwise, they should forget the system when not needed. The first important step was to understand how to organize the needed information in small screens on PDAs. This is a critical issue, as the operator should quickly access a lot of information, whereas small screens could cause a lot of steps needed to do this. Moreover, during an action it is usual that an actor, whose attention is completely concentrated on the task assigned to her/him, forgets the exact arrangement of the information items on the screen. Therefore, s/he should be able to recover in her/his mind the arrangement of the items through a fast glance to the PDA screen.

As it has been already widely studied and demonstrated (see for instance [46]), the maximum number of items that a subject can store instantaneously in a reduced time (approximately 200-300 milliseconds) is about 4 items. These considerations have brought us to divide the available information in 2 or 3 macro-categories. These categories can be easily accessed through the use of tabs on the left side of the screen(see Figure 7.2), without filling the screen with a huge set of objects.

Every macro-category is characterized by a different color so that the user gets easier to memorize and locate the context where s/he is (i.e. green color suggest the context of Communication). Each color is also highly contrasting in order to be clearly visible in particular light conditions (e.g. in night missions). Moreover, since the actors will use directly their fingers to access the screen features, this kind of interface makes it easier choosing the information with a low probability to push the wrong tab. When a tab is selected, it shows only the essential information about the context. It emerges from several studies that the users, when interacting with the systems, tend not to read whole words but only some letters in order to grasp the word meaning. Thus, the designers should draw interfaces using familiar formats and fonts, which must also be big enough. This allows the users to recall the words easier. This is specifically true in scenarios like disaster management where users should recognize the objects as quickly as possible, in order to minimize the response time to deal with rescue operations.

Finally, information about a specific issue can be logically reached as the user expects. For instance, see Figure 2a: if the user wants to see the map concerning the affected area, s/he has to select the tab "Mission" and push the button "See Geographic Information". Another important step has been to understand how to

capture the attention of the users while they were actually carrying out the tasks. A simple but effective idea has been to use sonorous pop-ups whenever the Work-list handlers request the direct user intervention. In order to avoid the continuous interaction with the system during the communication, as described below, the system allows members of the same team to interact with each other by an audio/textual communication, acting as a transceiver.

7.1.2 Interface Layout and Usage

This section presents the functionalities and actual usage of the user interface. The team leader's PDA has 3 tabs, whereas the PDAs of other members get only 2 tabs. Indeed, the generic team member has less functionalities than the leader.

When an emergency occurs, the Back-End sends emergency information to the team leader. It loads this information, which includes a description of the disaster, goals, and geographic information, onto the team leader's PDA (Figure 7.2a). When the team leader presses the "see geographic information" button, a map of the stricken place opens, showing the PDA owner's current position. To determine which members are best qualified to face the emergency, the team leader can perform a search that returns a list of actors who are ready to intervene at that moment, along with their personal capabilities (Figure 7.2b). Alternatively, the system can propose a team configuration based on available services deployed on the reachable team members, and automatically discovered.

Finally, the leader defines the process schema by customizing predefined generic process templates provided by the GUI (a screenshot is already shown in Figure 6.1), and moves the process schema to the PMS as input together with the initial context. From our collection of user requirements, we learned that, for the most part, emergency management operators define the processes to be enacted starting from predefined templates, and later instantiate the processes to the specific context. That is, there exist specific well known practices that are generically applicable. The PMS automatically assigns the tasks whose conditions are fulfilled to the actors able to execute them (because each task requires a well-defined set of capabilities), and each client's worklist handler receives notification of the assigned tasks (Figure 7.2c).

Next, the actor picks a task from his or her worklist, and the worklist handler starts the applications needed to perform the task (for example, in Figure 7.2d, the task consists of taking a photo). After executing a task, an actor's system automatically alerts the team leader's PDA, which has a complete view of the situation (Figure 7.2e). At each moment, the PMS can analyze whether new tasks are assignable and, if so, assign these tasks to the device whose capabilities best match the situation. If an actor becomes disconnected due to movement, another PDA can act as a bridge. The system automatically assigns a PDA to follow the disconnected node to keep alive a multihop path among devices, and a pop-up notifies the corresponding operator.

Finally, each PDA is equipped with a communication mechanism (Figure 7.2f).

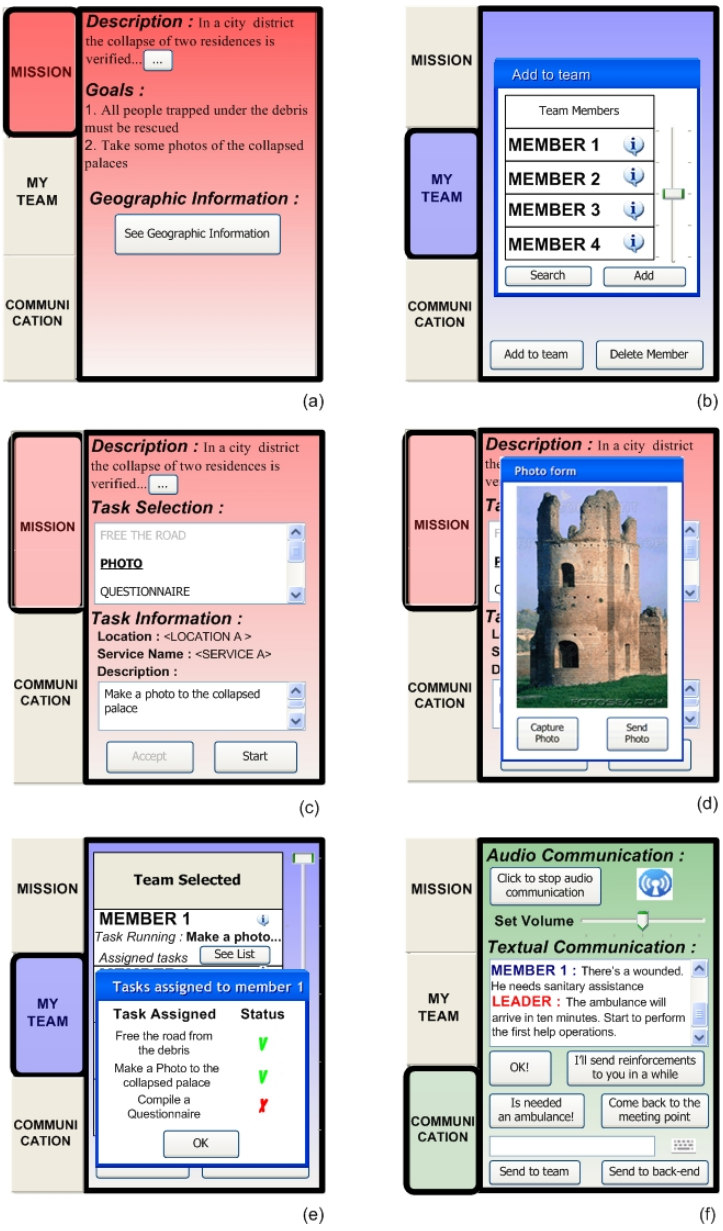


Figure 7.2: Screenshots of the GUI of the PDA Front-End

The system exploits the MANET technology to let team members interact through audio communication. In this way, the PDA also acts as a transceiver. If audio communication doesn't work (that is, the network is overloaded), the system provides the user with an alternative and lighter way to communicate - that is, exchange messages in a text form. As Figure 7.2f shows, users can add text to the form using a set of predefined messages or a virtual keyboard, as in a chat.

7.2 Implementation of the PMS engine

Nowadays, process management systems (PMSs [57]) are widely used for the management of "administrative" processes characterized by clear and well-defined structure. Besides such scenarios, which present mainly static characteristics (i.e., deviations are not the rule, but the exception), PMSs can also be used in mobile and highly dynamic situations, for instance, to coordinate operators/devices/robots/sensors [53]. In the WORKPAD project, the PMS component has been realized to work in scenarios concerning the emergency management, with the purpose to coordinate the activities of emergency operators within teams. There, the members of a team are equipped with PDAs and are coordinated through a PMS residing on a leader device (usually an ultra-mobile laptop). In such a PMS, process schemas (in the form of Activity Diagrams) are defined, describing different aspects, such as tasks/activities, control and data flow, tasks assignment to services, etc. Every task gets associated to a set of conditions which ought to be true for the task to be performed; conditions are defined on control and data flow (e.g., a previous task has to be completed or a variable needs to be assigned a specific range of values). Devices communicate among themselves through ad hoc networks (i.e. MANET), and in order to carry on the whole process, such devices need to be continually connected to the PMS. However, this cannot be guaranteed: the environment is highly dynamic and the movement of nodes (that is, devices and the related operators) around the affected area to carry out assigned tasks can cause disconnections and, thus, unavailability of nodes. This means that in highly dynamic scenarios processes can be easily invalidated, since the execution environment may change continuously because of frequent unforeseeable events and, thus, the process cannot carried on. Because of all this, some type of process adaptivity is desirable in such scenarios. But what does "adaptivity" mean? Adaptivity can be seen as the ability of the PMS to reduce the gap from the virtual reality, the (idealized) model of reality that is used by the PMS to deliberate, and the physical reality, the real world with the actual values of conditions and outcomes. For instance in scenarios of emergency management, PMS may restructure by including a "*follow X*" task to be assigned to a certain service in order to avoid the *X*'s disconnection. The reduction of such gap requires sufficient knowledge of both kinds of realities (virtual and physical). Such knowledge, determined and harvested by the services performing the process' tasks, would allow the sensing of deviations and the adaptation of the process for the final goal. This requires the provision of intelligent support for the planning and enactment of complex processes, that allows to capture the knowledge about the dynamic context of a process.

In [37], a general framework has been proposed to address the task of automatically adapting a process when a gap is sensed between the virtual and physical realities. As discussed there, other systems aim at addressing the critical issue of automatic adaptation. Here, we report on our ongoing efforts to operationalize such framework by making use of the INDIGOLOG agent architecture [25] developed at the University of Toronto's Cognitive Robotics Group.

7.2.1 A General Framework

This subsection introduces some preliminary notions, namely Situation Calculus SitCalc and IndiGolog, that are used as proper formalisms to reason about processes and exogenous events. Therefore it is not meant to give an all-comprehensive and very formal introduction of the notions. It aims mostly at giving an overall insight to those who are not very expert on such topics.

Preliminaries

PMS uses the situation calculus (SitCalc) to formalize adaptation. The SitCalc is a logic formalism designed for representing and reasoning about dynamic domains [44].

We will not go over the situation calculus in detail [21]; we merely note the following components: there is a special constant S_0 used to denote the *initial situation*, namely that situation in which no actions have yet occurred; there is a distinguished binary function symbol do, where $do(a, s)$ denotes the successor situation to s resulting from performing the action a; relations (resp. functions) whose values vary from situation to situation, are called *fluents*, and are denoted by predicate (resp. function) symbols taking a situation term as their last argument. There is a special predicate $Poss(a, s)$ used to state that action a is executable in situation s.

For convenience, we abbreviate with $do([a_1, \ldots, a_{n-1}, a_n], s)$ the term $do(a_n, do(a_{n-1}, ..., do(a_1, s)))$, which denotes the situation obtained from s by performing the sequence of actions $a_1, \ldots, a_n$.

Within this language, we can formulate domain theories which describe how the world changes as the result of the available actions. Here, we use action theories of the following form:

- Axioms describing the initial situation, S_0.
- Action precondition axioms, one for each primitive action a, characterizing $Poss(a, s)$.
- Successor state axioms, one for each relational fluent F. The successor state axiom for a particular fluent F captures the effects and non-effect of actions on F and has the following form:

$$F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a, s) \tag{7.1}$$

  where $\Phi_F(\vec{x}, a, s)$ is a formula fully capturing the truth-value of fluent F on objects $\vec{x}$ when action a is performed in situation s ($\vec{x}$, a, and s are all the free variables in Φ_F).
- Unique names axioms for the primitive actions.
- A set of foundational, domain independent axioms for situations Σ as in [44].

Construct	Meaning
a	primitive action
$\phi?$	wait for a condition
$\delta_1; \delta_2$	sequence
$\delta_1 \mid \delta_2$	nondeterministic branch
$\pi\, x.\, \delta$	nondeterministic choice of argument
δ^*	nondeterministic iteration
if ϕ **then** δ_1 **else** δ_2	conditional
while ϕ **do** δ	while loop
$\delta_1 \parallel \delta_2$	concurrency with equal priority
$\delta^{\parallel}$	concurrent iteration
$\langle \phi \rightarrow \delta \rangle$	interrupt
proc $P(\vec{x})$ δ **endProc**	procedure definition
$P(\vec{\theta})$	procedure call
$\Sigma(\delta)$	search operator

Table 7.1: IndiGolog constructs

A certain formula is *uniform in situation* s if s is the only situation term that appears in it. Sometimes, we use *situation-suppressed* formulas; these are uniform formulas with situation arguments suppressed (e.g. G denotes the situation-suppressed expression for $G(s)$). Finally, we can introduce an ordering among situations:

$$s \leq s' \Leftrightarrow \exists [a_1, \ldots, a_{n-1}, a_n].do([a_1, \ldots, a_{n-1}, a_n], s) = s'$$

On top of these theories of action, one can define complex control behaviors by means of high-level programs expressed in Golog-like programming languages. Specifically we focus on IndiGolog, which provides a set of programming constructs sufficient for defining every well-structured process as defined in [32].

IndiGolog is a logic-based languages born to program the behaviour of intelligent agents and robots. It derives from ConGolog to which it adds basically the lookahead *search operator*. Such operator allows to simulate the execution of a process with the aim of searching for a successful termination before actually performing the program in the real world. In its, turn ConGolog extends the original Golog by introducing construct for current execution of different operations. Table 7.1 summarizes the constructs of IndiGolog used in this thesis.

In the first line, a stands for a situation calculus action term whereas, in the second line, ϕ stands for a formula over situation calculus predicates including fluents, which are, then, evaluated in the current situation when IndiGolog program execution reaches ϕ.

The constructs listed included some nondeterministic ones. These include $(\delta_1 \mid \delta_2)$, which nondeterministically chooses between programs δ_1 and δ_2, $\pi\, x.\, \delta$, which nondeterministically picks a binding for the variable x and performs the program δ for this binding of x, and δ^*, which performs δ zero or more times. $\pi\, x_1, \ldots, x_n.\, \delta$

is an abbreviation for $\pi\, x_1 . \ldots . \pi\, x_n\, \delta$.

The constructs **if** ϕ **then** δ_1 **else** δ_2 and **while** ϕ **do** δ are the synchronized versions of the usual if-then-else and while-loop. They are synchronized in the sense that testing the condition ϕ does not involve a transition per se: the evaluation of the condition and the first action of the branch chosen are executed as an atomic unit. So these constructs behave in a similar way to the test-and-set atomic instructions used to build semaphores in concurrent programming.

We also have constructs for concurrent programming. In particular $(\delta_1 \parallel \delta_2)$ expresses the concurrent execution (interpreted as interleaving) of the programs δ_1 and δ_2. Observe that a program may become blocked when it reaches a primitive action whose preconditions are false or a wait action $\phi?$ whose condition ϕ is false. Then, execution of $(\delta_1 \parallel \delta_2)$ may continue provided another program executes next. Another concurrent programming construct is $(\delta_1 \rangle\!\rangle \delta_2)$, where δ_1 has higher priority than δ_2, and δ_2 may only execute when δ_1 is done or blocked. Finally, an interrupt $\langle \phi \rightarrow \delta \rangle$ has a trigger condition ϕ, and a body δ. If the interrupt gets control from higher priority processes and the condition ϕ is true, the interrupt triggers and the body is executed. Once the body completes execution, the interrupt may trigger again. $\langle \vec{x} : \phi \rightarrow \delta \rangle$ is an abbreviation for $\langle \exists \vec{x}.\phi \rightarrow \pi \vec{x}.\delta \rangle$.

Finally, the *search operator* $\Sigma(\delta)$ is used to specify that lookahead should be performed over the (nondeterministic) program δ to ensure that nondeterministic choices are resolved in a way that guarantees its successful completion.

Formally two predicates are introduced to specify program transitions:

- $Trans(\delta', s', \delta'', s'')$, given a program δ' and a situation s', returns *(i)* a new situation s'' resulting from executing a single step of δ', and *(ii)* δ'' which is the remaining program to be executed.

- $Final(\delta', s')$ returns true when the program δ' can be considered successfully completed in situation s'.

By using $Trans$ and $Final$ we can define a predicate $Do(\delta, s, s')$ which, given the starting situation s and a program δ, holds for all possible situations s' that result from executing δ starting from s such that situations s' are final with respect to program δ' remaining to execute. Formally:

$$Do(\delta, s, s') \Leftrightarrow \exists \delta'.Trans^*(\delta, s, \delta', s') \wedge Final(\delta', s')$$

where $Trans^*$ is the definition of the reflective and transitive closure of *Trans*. Notice that there may be more than one resulting situation s' since IndiGolog programs can be non-deterministic (e.g., due to concurrency).

To cope with the impossibility of backtracking actions executed in the real world, IndiGolog incorporates a new programming construct, namely the *search operator*. Let δ be any IndiGolog program, which provides different alternative executable actions. When the interpreter encounters program $\Sigma(\delta)$, before choosing among alternative executable actions of δ, it performs reasoning in order to

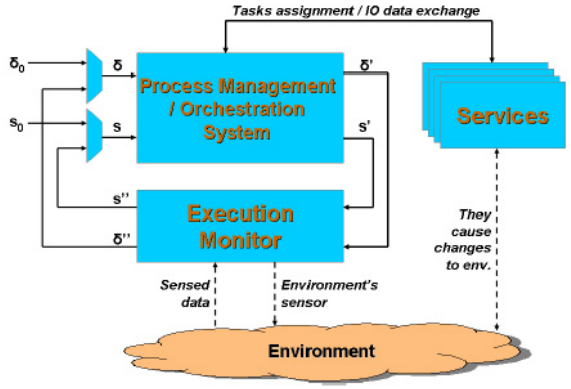


Figure 7.3: Execution Monitoring

decide for a step which still allows the rest of δ to terminate successfully. More precisely, according to [20], the semantics of the search operator is that

$$Trans(\Sigma(\delta), s, \Sigma(\delta'), s') \equiv Trans(\delta, s, \delta', s') \wedge \exists s^*.Do(\delta', s', s^*).$$

If δ is the entire program under consideration, $\Sigma(\delta)$ emulates complete offline execution.

Finally, our adaptation procedure will make use of *regression* (see [7] and [44]). Let $\varphi(do([a_1, \ldots, a_n], s))$ be a SitCalc formula with situation argument $do([a_1, \ldots, a_n], s)$. Then, $\mathcal{R}^s(\varphi(do([a_1, \ldots, a_n], s)))$ is the formula with situation s which denotes the facts/properties that must hold before executing $a_1, \ldots, a_n$ in situation s for $\varphi(do([a_1, \ldots, a_n], s))$ to hold (aka the weakest preconditions for obtaining φ). To compute the regressed formula $\mathcal{R}^s(\varphi(do([a_1, \ldots, a_n], s)))$ from $\varphi(do([a_1, \ldots, a_n], s))$, one iteratively replaces every occurrence of a fluent with the right-hand side of its successor state axiom [Formula 7.1] until every atomic formula has a situation argument that is simply s.

An Overview

The general framework is based on execution monitoring formally represented in SitCalc [36, 27]. After each action, the PMS has to align the internal world representation (i.e., the virtual reality) with the external one (i.e., the physical reality), since they could differ due to unforeseen events.

When using IndiGolog for process management, tasks are considered as predefined sequences of actions (see later) and processes as IndiGolog programs.

Before a process starts to be executed, the PMS takes the initial context from the real environment as initial situation, together with the program (i.e. the process) δ_0 to be carried on. The initial situation s_0 is given by first-order logic predicates. For each execution step, the PMS, which has a complete knowledge of the internal world (i.e., its virtual reality), assigns a task to a service. The only assignable tasks

are those ones whose preconditions are fulfilled. A service can collect from the PMS the data which are required in order to execute the task. When a service finishes executing the task, it alerts the PMS of its completion.

The execution of the PMS can be interrupted by the monitor when a misalignment between the virtual and the physical reality is sensed. When this happens, the monitor adapts the program to deal with such a discrepancy.

Figure 7.3 illustrates such an execution monitoring. At each step, PMS advances the process δ in the situation s by executing an action, resulting in a new situation s' with the process δ' remaining to be executed. The state[1] is represented as first-order formulas that are defined on situations. The current state corresponds to the boolean values of these formulas evaluated on the current situation.

Both the situation s' and the process δ' are given as input to the monitor. The monitor collects data from the environment through *sensors* (here *sensor* is any software or hardware component enabling to retrieve contextual information). If a deviation is sensed between the virtual reality as represented by s' and the physical reality as s'', the PMS internally generates a discrepancy $\vec{e} = (e_1, e_2, \ldots, e_n)$, which is a sequence of actions called *exogenous events* such that $s'' = do(\vec{e}, s')$.[2]

Notice that the process δ' may fail to be correctly executed (i.e., by assigning all tasks as required) in s''. If so, the monitor adapts the process by generating a new process δ'' that pursues at least each δ''s goal and is executable in s''. At this point, the PMS is resumed and the execution is continued from δ'' and s''.

We end this section by introducing our running example, stemming from the WORKPAD project.

Example 7.1. *The example is meant to code a possible process for managing the aftermath of an earthquake: a team is sent to the affected area to make an assessment, which comprises taking some valuable photos, compiling a questionnaire and sending all these data to the headquarter (see Figure 7.4). Here we assume that it is already known which buildings have to be assessed, namely buildings A, B and C. The team is equipped with PDAs in which some software services are installed and members are communicating with each other through a* MANET *network. For each building, an actor compiles a questionnaire by using a certain software service, that is an specific application installed on some actor devices. Compiling questionnaires can be done anywhere: that is, no movement is required. Then, another actor/service has to be sent to the specific building to collect some pictures (this, conversely, requires movement). Finally, according to the information filled in the questionnaire, a third actor/service evaluates effectiveness of collected pictures. In order to evaluate properly the pictures, a certain minim number of pictures is required as input: if less, the evaluation cannot be done. Once evaluated, if pictures are judged as not effective, the task of taking new pictures is scheduled*

[1] Here we refer as *state* both the tasks' state (e.g, performable, running, terminated, etc.) and the process' variables. The use of the latter variables are twofold: from the one hand, the routing is defined on them and, from the other hand, they allow to learn when a task may fire.

[2] Note that the action sequence $\vec{e}$ might not be the one that really occurred.

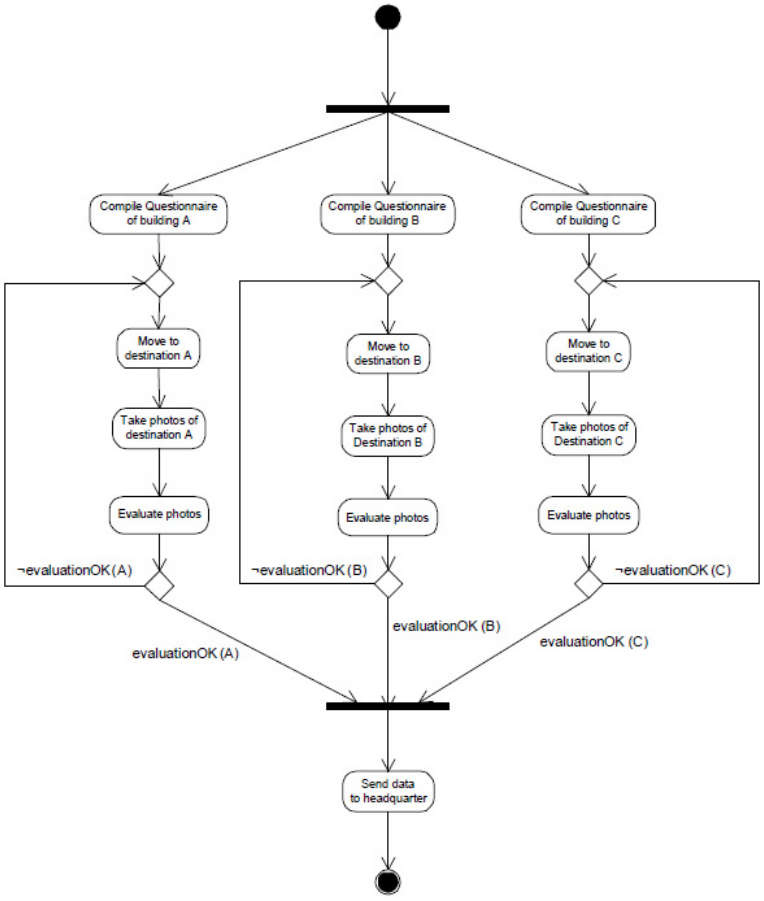


Figure 7.4: A possible process to be carried on in disaster management scenarios

again (as well as the evaluation of the new pictures). When these steps have been performed for the three buildings A, B and C, the collected data (questionnaires and pictures) are sent to the headquarter.

Coordination and data exchange requires MANET *nodes to be continually connected each other. But this is not guaranteed in a* MANET. *The environment is highly dynamic, since nodes move in the affected area to carry out assigned tasks. Movements may cause possible disconnections and, so, unavailability of nodes, and, consequently, unavailability of provided services. Therefore processes should be adapted, not simply by assigning tasks in progress to other services, but also considering possible recovery of the services.* □

7.2.2 Process Formalisation in Situation Calculus

Next we detail the general framework proposed above by using Situation Calculus and IndiGolog. We use some domain-independent predicates to denote the various objects of interest in the framework:

- $service(a)$: a is a service
- $task(x)$: x is a task
- $capability(b)$: b is a capability
- $provide(a, b)$: the service a provides the capability b
- $require(x, b)$: the task x requires the capability b

Every task execution is the sequence of four PMS actions: *(i)* the assignment of the task to a service, resulting in the service being not free anymore; *(ii)* the notification to the service to start executing the task. Then, the service carries out the tasks and, after receiving the service notification of the task conclusion, *(iii)* the PMS acknowledges the successful task termination. Finally, *(iv)* the PMS releases the service, which becomes free again. We formalise these four actions as follows:

- $Assign(a, x)$: task x is assigned to a service a
- $Start(a, x, p)$: service a is allowed to start the execution of task x. The input provided is p.
- $AckTaskCompletion(a, x)$: service a concluded successfully the executing of x.
- $Release(a, x)$: the service a is released with respect to task x.

In addition, services can execute two actions:

- $readyToStart(a, x)$: service a declares to be ready to start performing task x
- $finishedTask(a, x, q)$: services declares to have completed the execution of task x returning output q.

The terms p and q denote arbitrary sets of input/output, which depend on the specific task. Special constant $\emptyset$ denotes empty input or output. The interleaving of actions performed by the PMS and services is as follows. After the assignment of a certain task x by $Assign(a, x)$, when the service a is ready to start executing, it executes action $readyToStartTask(a, x)$. At this stage, PMS executes action $Start(a, x, p)$, after which a starts executing task x. When a completes task x, it executes the action $finishedTask(a, x, q)$. Specifically, we envision that actions $finishedTask(\cdot)$ are those in charge of changing properties of world as result of

executing tasks. When x is completed, PMS is allowed in any moment to execute sequentially the actions $AckTaskCompletion(a, x)$ and $Release(a, x)$. The program coding the process will the executed by only one actor, specifically the PMS. Therefore, actions $readyToStartTask(\cdot)$ and $finishedTask(\cdot)$ are considered as external and, hence, not coded in the program itself.

For each specific domain, we have several fluents representing the properties of situations. Some of them are modelled independently of the domain whereas others, the majority, are defined according to the domain. If they are independent of the domain, they can be always formulated as defined in this chapter. Among the domain-independent ones, we have fluent $free(a, s)$, that denotes the fact that the service a is free, i.e., no task has been assigned to it, in the situation s. The corresponding successor state axiom is as follows:

$$\begin{aligned} &free(a, do(t, s)) \Leftrightarrow \\ &\quad \big(\forall x.t \neq Assign(a, x) \wedge free(a, s)\big) \vee \\ &\quad \big(\neg free(a, s) \wedge \exists x.t = Release(a, x)\big) \end{aligned} \tag{7.2}$$

This says that a service a is considered free in the current situation if and only if a was free in the previous situation and no tasks have been just assigned to it, or a was not free and it has been just released. There exists also the domain-independent fluent $enabled(x, a, s)$ which aims at representing whether service a has notified to be ready to execute a certain task x so as to enabled it. The corresponding successor-state axiom:

$$\begin{aligned} &enabled(x, a, do(t, s)) \Leftrightarrow \\ &\quad \big(enabled(x, a, s) \wedge \forall q.t \neq finishedTask(a, x, q)\big) \vee \\ &\quad \big(\neg enabled(x, a, s) \wedge t = readyToStartTask(a, x)\big) \end{aligned} \tag{7.3}$$

This says that $enabled(x, a, s)$ holds in the current situation if and only if it held in the previous one and no action $finishedTask(a, x, q)$ has been performed or it was false in the previous situation and $readyToStartTask(a, x)$ has been executed. This fluent aims at enforcing the constraints that the PMS can execute $Start(a, x, p)$ only after a performed $begun(a, x)$ and it can execute $AckTaskCompletion(a, x, q)$ only after $finishedTask(a, x, q)$.

This can represented by two pre-conditions on actions $Start(\cdot)$ and $AckTaskCompletion(\cdot)$:

$$\begin{aligned} &\forall p.Poss(Start(a, x, p), s) \Leftrightarrow enabled(x, a, s) \\ &\forall p.Poss(AckTaskCompletion(x, a), s) \Leftrightarrow \neg enabled(x, a, s) \end{aligned} \tag{7.4}$$

provided that $AckTaskCompletion(x, a)$ never comes before $Start(x, a, p), s$.

Furthermore, we introduce a domain-independent fluent $started(x, a, p, s)$ that holds if and only if an action $Start(a, x, p)$ has been executed but the dual $AckTaskCompletion(x, a)$ has not yet:

$$\begin{aligned} &started(a, x, p, do(t, s)) \Leftrightarrow \\ &\quad \big(started(a, x, p, s) \wedge t \neq Stop(a, x)\big) \vee \\ &\quad \big(\nexists p'.started(x, a, p', s) \wedge t = Start(a, x, p)\big) \end{aligned} \tag{7.5}$$

In addition, we make use, in every specific domain, of a predicate called $available(a, s)$ which denotes whether a service a is available in situation s for tasks assignment. However, $available$ is domain-dependent and, hence, requires to be defined specifically for every domain. Knowing whether a service is available is very important for the PMS when it has to perform assignments. Indeed, a task x is assigned to the best service a which is available and provides every capability required by x. The fact that a certain service a is free does not imply it can be assigned to tasks (e.g., in the example described above it has to be free as well as it has to be indirectly connected to the coordinator). The definition of $available(\cdot)$ must enforce the following condition:

$$\forall a\ s.available(a, s) \Rightarrow free(a, s) \tag{7.6}$$

We do not give explicitly pre-conditions to task. We assume tasks can always be executed. We assume that, given a task, if some conditions do not hold, then the outcomes of that tasks are not as expected (in other terms, it fails).

We illustrate such notions on our running example.

Example 7.1 (cont.). *We formalize the scenario in Example 7.1:*

- $at(loc, p, s)$ *is true if service* w *is located at coordinate* $loc = \langle loc_x, loc_y, loc_z \rangle$ *in situation* s*. In the starting situation* S_0*, for each service* a_i*, we have* $at(a_i, loc_i, S_0)$ *where location* loc_i *is obtained through GPS sensors.*

- $evaluationOK(loc, s)$ *is true if the photos taken are judged as having a good quality, with* $evaluationOK(loc, S_0) =$ `false` *for each location* loc*.*

- $infoSent(s)$ *is true in situation* s *if the information concerning injured people at destination* d *has been successfully forwarded to the headquarter. There holds* $infoSent(d, S_0) =$ `false`*.*

- $photoBuild(loc, n, s)$ *is true if in location* loc n *photos have been taken. In the starting situation* S_0 $photoBuildA(loc, 0, S_0)$ *for all locations* loc*.*

Before giving the success-state axioms for the above fluents, we define some abbreviations:

- $available(a, s)$*: which states a service* a *is available if it is connected to the coordinator device (denoted by* `Coord`*) and is free.*

- $connected(w, z, s)$*: which is true if in situation* s *the services* w *and* z *are connected through possibly multi-hop paths.*

- $neigh(w, z, s)$*: which holds if the services* w *and* z *are in radio-range in the situation* s*.*

Their definitions are as follows:

$$neigh(w_0, w_1, s) \stackrel{def}{=} at(w_0, p_0, s) \wedge at(w_1, p_1, s) \wedge \parallel p_0 - p_1 \parallel < rrange$$

$$\begin{aligned} connected(w_0, w_1, s) \stackrel{def}{=}\ & neigh(w_0, w_1, s) \\ & \vee \big(\exists w_2.neigh(w_0, w_2, s) \wedge neigh(w_2, w_1, s)\big) \\ & \vee \big(\exists w_2, w_3.neigh(w_0, w_2, s) \wedge neigh(w_2, w_3, s) \wedge neigh(w_3, w_1, s)\big) \\ & \vee \big(\exists w_2, w_3, w_4 \ldots\big) \\ & \vee \ldots \\ & \vee \big(\exists w_2, w_3, \ldots, w_n.neigh(w_0, w_2, s) \wedge neigh(w_2, w_3, s) \\ & \quad \wedge neigh(w_3, w_1, s) \wedge \ldots\big) \end{aligned}$$

$$available(w, s) \stackrel{def}{=} free(w, s) \wedge connected(w, \texttt{Coord}, s))$$

The successor state axioms for this domain are:

$$\begin{aligned} & at(a, loc, do(t, s)) \Leftrightarrow \\ & \quad \big(at(a, loc, s) \wedge \forall loc'.t \neq finishedTask(a, \textsf{Go}, loc')\big) \\ & \quad \vee \big(\neg at(a, loc, s) \wedge t = finishedTask(a, \textsf{Go}, loc) \wedge started(a, \textsf{Go}, loc, s)\big) \end{aligned}$$

$$\begin{aligned} & evaluationOK(loc, do(t, s)) \Leftrightarrow evaluationOK(loc, s) \\ & \quad \vee \big(\exists a.t = finishedTask(a, \textsf{Evaluate}, \langle loc, \texttt{OK}\rangle) \\ & \qquad \wedge photoBuild(loc, n, s) \wedge \exists p.started(a, \textsf{Evaluate}, p, s) \wedge n \geq \texttt{threshold}\big) \end{aligned}$$

$$\begin{aligned} & infoSent(do(t, s)) \Leftrightarrow infoSent(s) \\ & \quad \vee \big(\exists a.t = finishedTask(a, \textsf{SendToHeadquarter}, \texttt{OK}) \\ & \wedge \exists p.started(a, \textsf{SendToHeadquarter}, p, s)\big) \end{aligned}$$

$$\begin{aligned} & photoBuild(loc, n, do(t, s)) \Leftrightarrow \\ & \quad \big(\exists a, m, o.photoBuild(loc, m, s) \wedge t = finishedTask(a, \textsf{TakePhoto}, \langle loc, o\rangle) \\ & \qquad \wedge n = m + o \wedge at(a, loc, s) \wedge \exists p.started(a, \textsf{TakePhoto}, p, s)\big) \\ & \quad \vee \big(\exists a, o.photoBuild(loc, n, s) \wedge t = finishedTask(a, \textsf{TakePhoto}, \langle loc, o\rangle) \\ & \qquad \wedge \neg at(a, loc, s) \wedge \exists p.started(a, \textsf{TakePhoto}, p, s)\big) \\ & \quad \vee \big(\forall a, o.photoBuild(loc, n, s) \wedge t \neq finishedTask(a, \textsf{TakePhoto}, \langle loc, o\rangle)\big) \end{aligned}$$

It is worthy noting that all fluents which denote world properties of interest are changed by $finishedTask$, as already told. Moreover, the value of fluent $photoBuild(loc, n, s)$ is updated by the execution of task Go *only if the executor is at location loc. Otherwise, the photos taken are not considered as valuable. Even if that is not formally the pre-condition of the task (the task can be executed in any case), in fact that is a condition that has to hold in order the task be executed.*

□

```
MAIN()
 1  (EVALTAKE(LocA) || EVALTAKE(LocB) || EVALTAKE(LocC));
 2  π.a0[available(a0) ∧ ∀c.require(c, SendByGPRS) ⇒ provide(a0, c)];
 3    Assign(a0, SendByGPRS);
 4    Start(a0, SendByGPRS, ∅);
 5    AckTaskCompletion(a0, SendByGPRS);
 6    Release(a0, SendByGPRS);

EVALTAKE(Loc)
 1  π.a1[available(a1) ∧ ∀c.require(c, CompileQuest) ⇒ provide(a1, c)];
 2    Assign(a1, CompileQuest);
 3    Start(a1, CompileQuest, Loc);
 4    AckTaskCompletion(a1, CompileQuest);
 5    Release(a1, CompileQuest);
 6  while ¬evaluationOK(Loc)
 7  do
 8     π.a2[available(a2) ∧ ∀c.require(c, Go) ⇒ provide(a2, c)];
 9       Assign(a2, Go);
10       Start(a2, Go, Loc);
11       AckTaskCompletion(a2, Go);
12       Start(a2, TakePhoto, Loc);
13       AckTaskCompletion(a2, TakePhoto);
14       Release(a2, TakePhoto);
15     π.a3[available(a3) ∧ ∀c.require(c, EvaluatePhoto) ⇒ provide(a3, c)];
16       Assign(a3, EvaluatePhoto);
17       Start(a3, EvaluatePhoto, Loc);
18       AckTaskCompletion(a3, EvaluatePhoto);
19       Release(a3, EvaluatePhoto);
```

Figure 7.5: The IndiGolog program corresponding to the process in Figure 7.4

7.2.3 The PMS implementation

This section is devoted to describe PMS, the concrete implementation of the framework described in the previous one. For this aim, we used in IndiGolog platform developed by University of Toronto in collaboration with the Agent Group of RMIT University, Melbourne.

The concrete implementation of PMS has encountered two main group of issues. Firstly, the IndiGolog platform was targeted to the agent and robot programming and, hence, using it for process management, which is not a close application field, has been quite difficult. For example, we needed the inclusion of the construct atomic to define a sequence of actions that have to be executed all together and cannot be interrupted and actions in sequences concurrently executing. Such a kind of construct makes less sense in the field of robots, where it is quite important in process management in order to introduce the concept of transaction. In fact, the development has been carried on with a tight collaboration with the conceivers and developers who very kindly made some changes to meet our requirements.

Secondly, the theoretical framework did not consider the features and limitations that are actually available in the platform. For instance, the theoretical framework supposed to stop the process and restructure it by placing the recovery beforehand. In practice, the platform does not allow to change the program that codes the process when already started. In order to overcome this limitation, we committed to use interrupts at different priorities.

7.2.4 The IndiGolog Platform

This section describes the IndiGolog-based platform that we have used to implement the framework described in Section 7.2.1[3]. Part of this section is a summary of the work published in [26] by kind agreement with its authors.

The agent platform to be described here is a *logic-programming* implementation of IndiGolog that allows the incremental execution of high-level Golog-like programs. This implementation of IndiGolog is modular and easily extensible so as to deal with any external platform, as long as the suitable interfacing modules are programmed.

Although most of the code is written in vanilla Prolog, the overall architecture is written in the well-known open source SWIProlog[4] [61]. SWIProlog provides flexible mechanisms for interfacing with other programming languages such as Java or C, allows the development of multi-threaded applications, and provides support for socket communication and constraints.

Generally speaking, the IndiGolog implementation provides an incremental interpreter of high-level programs as well as a framework for dealing with the *real* execution of these programs on *concrete* platforms or devices. This amounts to

[3] Available at `http://sourceforge.net/projects/indigolog/`.

[4] `http://www.swi-prolog.org/`

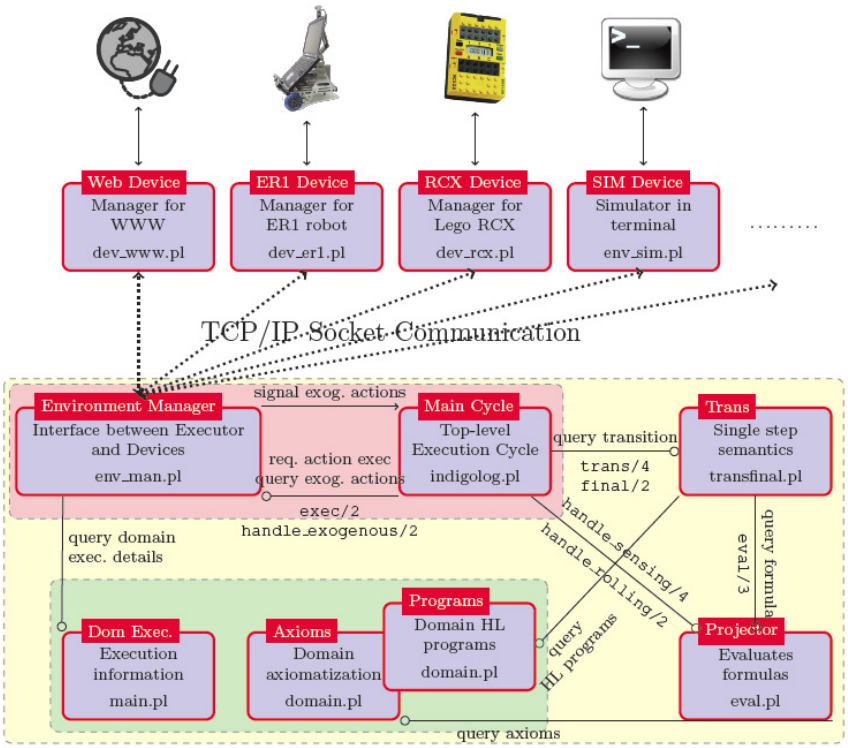


Figure 7.6: The IndiGolog implementation architecture. Links with a circle ending represent goal posted to the circled module (as from [26])

handling the real execution of actions on concrete devices (e.g., a real robot platform), the collection of sensing outcome information (e.g., retrieving some sensor's output), and the detection of exogenous events happening in the world. To that end, the architecture is modularly divided into six parts, namely, *(i)* the top-level main cycle; *(ii)* the language semantics; *(iii)* the temporal projector; *(vi)* the environment manager; *(v)* the set of device managers; and finally *(vi)* the domain application. The first four modules are completely domain independent, whereas the last two are designed for specific domain(s). The architecture is depicted in Figure 7.6.

The top-level main cycle and language semantics

It realizes the *sense-think-act* loop well-known in the agent community [34]:

1. check for exogenous events that have occurred;
2. calculate the next program step; and
3. if the step involves an action, *execute* the action.

The main predicate of the main cycle is `indigo/2`; a goal of the form `indigo(E,H)` states that the high-level program `E` is to be executed online at history `H`.

The first thing the main cycle does is to assimilate all exogenous events that have occurred since the last execution step. After all exogenous actions have been assimilated and the history progressed as needed, the main cycle goes on to actual executing the high-level program `E`. First, if the current program to be executed is terminating in the current history, then the top-level goal `indigo/2` succeeds (third clause). Otherwise, the interpreter checks whether the program can evolve a single step (fourth clause) by relaying on predicate `trans/4` (explained below). If the program evolves without executing any action, then the history remains unchanged and we continue to execute the remaining program from the same history. If, however, the step involves performing an action, then this action is executed and incorporated into the current history, together with its sensing result (if any), before continuing the execution of the remaining program.

As mentioned above, the top-level loop relies on two central predicates, namely, `final/2` and `trans/4`. These predicates implement relations $Trans$ and $Final$, giving the single step semantics for each of the constructs in the language. It is convenient, however, to use an implementation of these predicates defined over histories instead of situations. Indeed, the constructs of the IndiGolog interpreter never treat about situations but they are always assuming to work on the current situation. So, for example, these are the corresponding clauses for sequence (represented as a list), tests, nondeterministic choice of programs, and primitive actions:

```
final([E|L],H) :- final(E,H), final(L,H).
trans([E|L],H,E1,H1) :- final(E,H), trans(L,H,E1,H1).
trans([E|L],H,[E1|L],H1) :- trans(E,H,E1,H1).

final(ndet(E1,E2),H) :- final(E1,H) ; final(E2,H).
trans(ndet(E1,E2),H,E,H1) :- trans(E1,H,E,H1).
trans(ndet(E1,E2),H,E,H1) :- trans(E2,H,E,H1).

trans(?(P),H,[],H) :- eval(P,H,true).
trans(E,H,[],[E|H]) :- action(E), poss(E,P), eval(P,H,true).
/* Obs: no final/2 clauses for action and test programs */
```

These Prolog clauses are almost directly "lifted" from the corresponding axioms for $Trans$ and $Final$. Predicates `action/1` and `poss/2` specify the actions of the domain and their corresponding precondition axioms; both are defined in the domain axiomatization (see below). More importantly, `eval/3` is used to check the truth of a condition at a certain history, and is provided by the temporal projector, described next.

The naive implementation of the search operator would deliberate from scratch at every point of its incremental execution. It is clear, however, that one could do better than that, and cache the successful plan obtained and avoid planning in most cases:

```
final(search(E),H) :- final(E,H).
trans(search(E),H,path(E1,L),H1) :-
```

```
      trans(E,H,E1,H1), findpath(E1,H1,L).

/* findpath(E,H,L): solve (E,H) and store the path in list L  */
/* L = list of configurations (Ei,Hi) expected along the path */
findpath(E,H,[(E,H)]) :- final(E,H).
findpath(E,H,[(E,H)|L]) :- trans(E,H,E1,H1), findpath(E1,H1,L).
```

So, when a search block is solved, the whole solution path found is stored as the sequence of configurations that are expected. If the actual configurations match, then steps are performed without any reasoning (first `final/2` and `trans/4` clauses for program `path(E,L)`). On the other hand, if the actual configuration does not match the one expected next, for example, because an exogenous action occurred and the history thus changed, re-planning is performed to look for an alternative path (code not shown).

The temporal projector

The temporal projector is in charge of maintaining the agent's beliefs about the world and evaluating a formula relative to a history. The projector module provides an implementation of predicate `eval/3`: goal `eval(F,H,B)` states that formula `F` has truth value `B`, usually `true` or `false`, at history `H`.

Predicate `eval/3` is used to define `trans/4` and `final/2`, as the legal evolutions of high-level programs may often depend on what things are believed true or false.

We assume then that users provide definitions for each of the following predicates for fluent f, action a, sensing result r, formula w, and arbitrary value v:

- $fun\_fluent(f)$, f is a functional fluent;
- $rel\_fluent(f)$, f is a functional fluent;
- $prim\_action(a)$, a is a ground action;
- $init(f, v)$, initially, v is the value for fluent f in the starting situation;
- $poss(a, w)$, it is possible to execute action a provided formula w is known to be true;
- $causes\_val(a, f, v, w)$, action a affects the value of f:

Formulas are represented in Prolog using the obvious names for the logical operators and with all situations suppressed; histories are represented by lists of the form $o(a, r)$ where a represents an action and r a sensing result. We will not go over how formulas are recursively evaluated, but just note that there exists a predicate *(i)* `kTrue`(w, h) is the main and top-level predicate and it tests if the formula w is at history h. Finally, the interface of the module is defined as follows:

```
eval(F,H,true) :- kTrue(F,H).
eval(F,H,false) :- kTrue(neg(F),H).
```

The environment manager and the device managers

Because the architecture is meant to be used with concrete agent/robotic platforms, as well as with software/simulation environments, the online execution of IndiGolog programs must be linked with the external world. To that end, the *environment manager* (EM) provides a complete interface with all the external devices, platforms, and real-world environments that the application needs to interact with.

In turn, each external device or platform that is expected to interact with the application (e.g., a robot, a software module, or even a user interface) is assumed to have a corresponding *device manager*, a piece of software that is able to talk to the actual device, instruct it to execute actions, as well as gather information and events from it. The device manager understands the "hardware" of the corresponding device and provides a high-level interface to the EM. It provides an interface for the execution of actions (e.g., `assign`, `start`, etc.), the retrieval of sensing outcomes for actions, and the occurrence of exogenous events (e.g., `disconnect` as well as `finishedTask`).

Because actual devices are independent of the IndiGolog application and may be in remote locations, device managers are meant to run in different processes and, possibly, in different machines; they communicate then with the EM via TCP/IP sockets. The EM, in contrasts, is part of the IndiGolog agent architecture and is tightly coupled with the main cycle. Still, since the EM needs to be open to the external world regardless of any computation happening in the main cycle, the EM and the main cycle run in different (but interacting) threads, though in the same process and Prolog run-time engine.[5]

So, in a nutshell, the EM is responsible of executing actions in the real world and gathering information from it in the form of sensing outcome and exogenous events by communicating with the different device managers. Precisely, given a domain high-level action (e.g., $assign(WrkList, Srvc)$), the EM is in charge of: *(i)* deciding which actual "device" should execute the action; *(ii)* ordering its execution by the device via its corresponding device manager; and finally *(iii)* collecting the corresponding sensing outcome. To realize the execution of actions, the EM provides an implementation of `exec/2` to the top-level main cycle: `exec(A,S)` orders the execution of action `A`, returning `S` as its sensing outcome.

When the system starts, the EM starts up all device managers required by the application and sets up communications channels to them using TCP/IP stream sockets. Recall that each real world device or environment has to have a corresponding device manager that understands it. After this initialization process, the EM enters into a *passive mode* in which it asynchronously listens for messages arriving from the various devices managers. This passive mode should allow the top-level main cycle to execute without interruption until a message arrives from some device manager. In general, a message can be an exogenous event, a sensing outcome of some recently executed action, or a system message (e.g., a device

[5] SWIProlog provides a clean and efficient way of programming multi-threaded Prolog applications.

being closed unexpectedly). The incoming message should be read and handled in an appropriate way, and, in some cases, the top-level main cycle should be notified of the occurred event.

The domain application

From the user perspective, probably the most relevant aspect of the architecture is the specification of the domain application. Any domain application must provide:

1. An *axiomatization of the dynamics of the world*. Such axiomatization would depend on the temporal projector to be used.

2. One or more *high-level agent programs* that will dictate the different agent behaviors available. In general, these will be IndiGolog programs.

3. All the necessary *execution information* to *run* the application in the external world. This amounts to specifying which external devices the application relies on (e.g., the device manager for the ER1 robot), and how high-level actions are actually executed on these devices (that is, by which device each high-level action is to be executed). Information on how to translate high-level symbolic actions and sensing results into the device managers' low-level representations, and vice-versa, could also be provided.

7.2.5 The PMS

Figure 7.7 shows how conceptually PMS has been integrated into the IndiGolog interpreter.

At the beginning, we envision a responsible person to design the process specification through a Graphical Tool, namely SPIDE (Figure 7.8 shows a screen shot), which generates an accordant XML files [41]. Specifically, it is meant to generate the XML specification file which should contain a formal domain theory as well as the process schema and the action conditions. It allows to define specific templates with a finite number of open options. When the instance needs to be created, an operator chooses the proper template from a repository and close the open points, thus transforming the abstract template in a concrete process specification.

The *XML-to-IndiGolog Parser* component translates a SPIDE's XML specification in three conceptual parts:

Domain Program. The IndiGolog program corresponding to the designed process. It includes also some helper procedures to handle the task executions, the interaction with the external services and other features.

Domain Axioms It comprises the action theory: the set of fluents modeling world properties of interest, the set of available tasks, and the successor-state axioms which describes how the actions applied on tasks change the fluents.

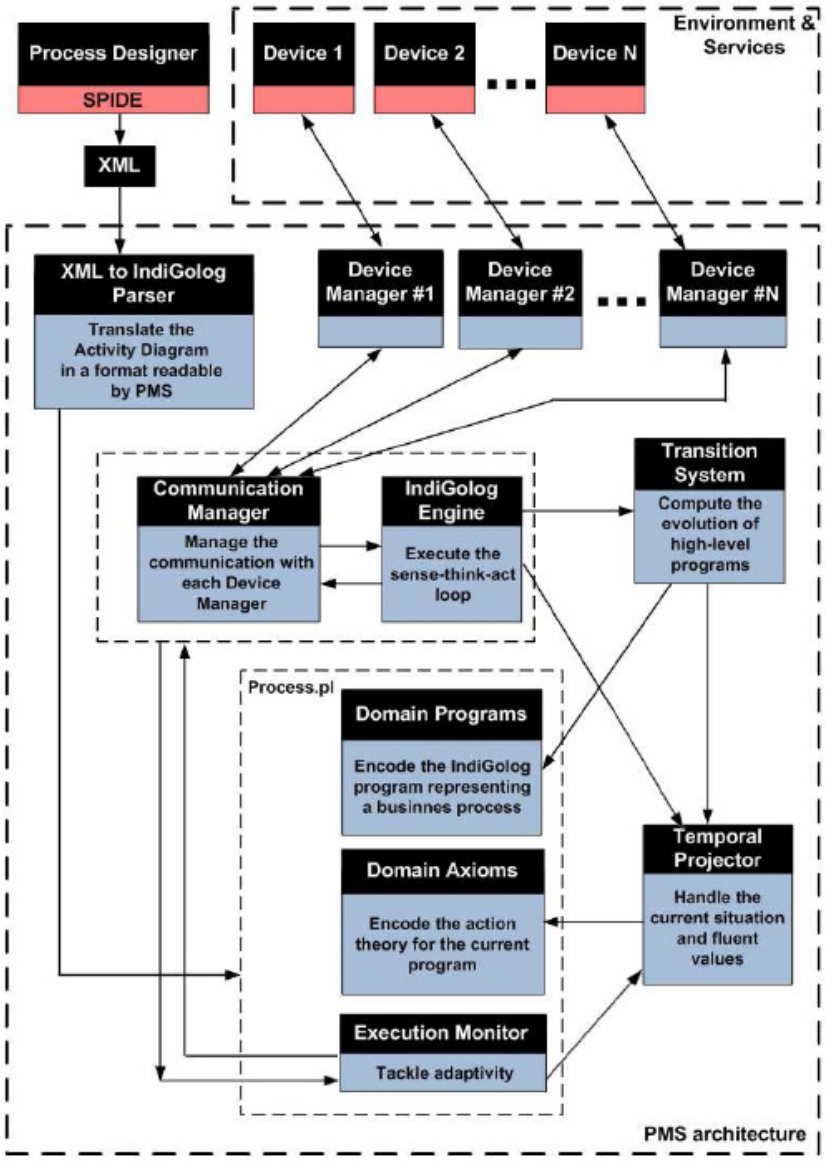


Figure 7.7: Architecture of the PMS.

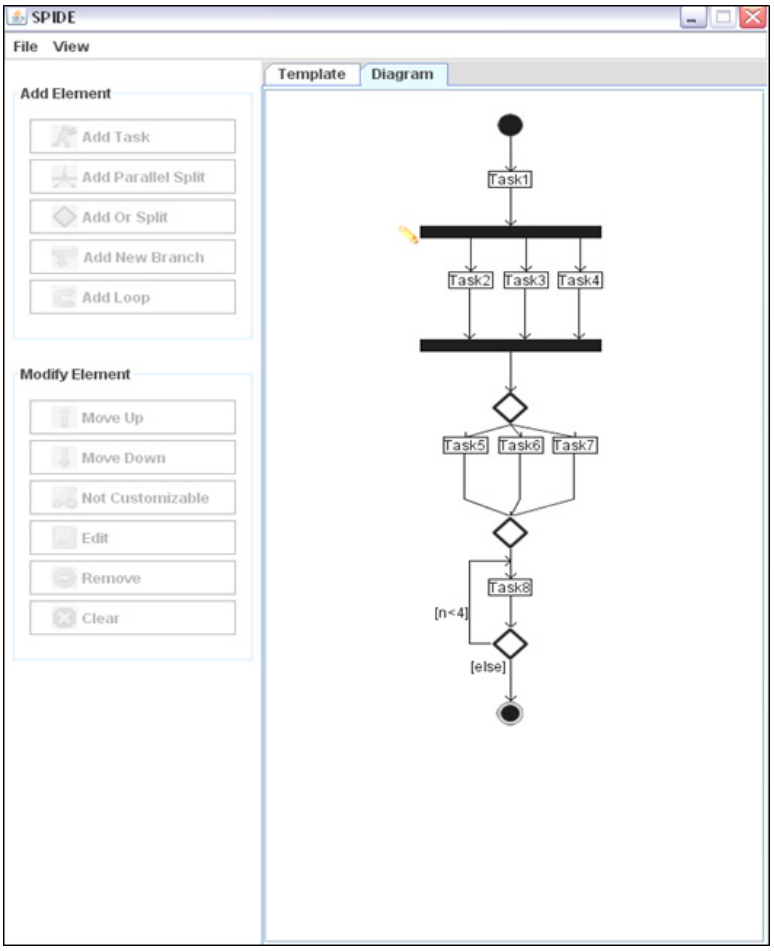


Figure 7.8: The SPIDE Tool

Some axiomatization parts are, in fact, independent of the domain, and, hence, remain unchanged when passing from a domain to another. On the contrary, other axioms are modeled according to the domain and model how domain-dependent fluents change as result of the task executions.

Execution Monitor This parts is always generated in the same way and does not take the specific domain into account.

When the program is translated in the Domain Program and Axioms, the *Communication Manager* (CM) starts up all of *device managers*, which are basically some drivers for making communicate PMS with the services and sensors installed on devices. PMS holds a device manager for each device hosting services. After this initialization process, CM activates the *IndiGolog Engine*, which is in charge of executing IndiGolog programs by realising the main cycle described in subsection 7.2.4. Then, CM enters into a passive mode where it is is listening for messages arriving from the devices through the device managers. In general, a message can be a exogenous event harvested by a certain sensor installed on a given device as well as a message notifying the beginning or the completion of a certain task.

The Communication Manager can be invoked by the IndiGolog Engine whenever it produces an action for execution. The IndiGolog Engine relies on two further modules named *Transition System* and *Temporal Projector*. The former is used to compute the evolution of IndiGolog programs according to the statements' semantic., whereas the latter is in charge of holding the current situations throughout the execution, making possible to evaluate the fluent values.

From the one side, the *Execution Monitor* makes use of CM which notifies which notifies the occurrence of exogenous events; from the other side, it relies on the *Temporal Projector* to get the updated values of fluents.

Coding processes by the IndiGolog interpreter

This sub-section turns to describe how processes can be concretely coded as IndiGolog programs by using the interpreter described in sub-section 7.2.4. Interested readers may look at Example 7.2, which shows the most significant parts of the interpreter code.

The process requires a model definition for the predicates that are defined in sub-section 7.2.2: $service(a)$, $task(x)$, $capability(b)$, $provide(a, b)$, $require(x, b)$. In addition, we introduced predicate $identifiers(i)$, which defines the valid identifiers for tasks. Indeed, the process specification may comprise certain tasks more than once; of course, different instances of the same task have to be distinguished as they are different pieces of work.

Example 7.2. *The following is the code of the IndiGolog interpreter giving a definition of the aforementioned predicates for the running example. Specifically, the example assumes the team to be composed by five services, all humans, that are*

univocally identified by a number. Predicate `domain(N,X)` *has been made available by the* IndiGolog *interpreter itself. And it holds whether element* N *is into list* X.

```
/* Available services */
services([1,2,3,4,5]).
service(Srvc) :- domain(Srvc,services).

/* Tasks defined in the process specification */
tasks([TakePhoto,EvaluatePhoto,CompileQuest,Go,SendByGPRS]).
task(Task) :- domain(Task,tasks).

/* Capabilities relevant for the process of interest*/
capabilities([camera,compile,gprs,evaluation]).
capability(B) :- domain(B,capabilities).


/* The list of identifiers that may be used
to distinguish different instances of the same task*/
task_identifiers([id_1,id_2,id_3,id_4,id_5,id_6,id_7,id_8,id_9,
id_10,id_11,id_12,id_13,id_14,id_15,id_16,id_17,id_18,id_19,
id_20]). id(D) :- domain(D,task_identifiers).

/* The capabilities required for each task */
required(TakePhoto,camera).
required(EvaluatePhoto,evaluation).
required(CompileQuest,compile).
required(SendByGPRS,gprs).

/The capabilities provided by each service */

provide(1,gprs).
provide(1,evaluation).
provide(2,compile).
provide(2,evaluation).
provide(2,camera).
provide(3,compile).
provide(4,evaluation).
provide(4,camera).
provide(5,compile).
```

□

Tasks with their identifiers and inputs are packaged into elements $workitem(Task, Id, Input)$ of predicates $listElem(workitem)$. The work-item element can be grouped in lists identified by elements $worklist(\cdot)$. The following is the corresponding Prolog code:

```
worklist([]).
worklist([ELEM | TAIL]) :- worklist(TAIL),listelem(ELEM).
```

Indeed, actions $assign(\cdot)$ and $release(\cdot)$ take as input elements $worklist(\cdot)$. In fact, this implementation assigns one $worklist(\cdot)$ to one proper service that is capable to execute all tasks in the list. The assignment of lists of tasks to services

rather than single tasks is motivated by the fact that we are willing to constrain multiple tasks to be executed by the same service.

Example 7.2 (cont.). *The example shows the definition of the different types of valid work items and their input parameters. Specifically, the first definition of* $listElem$ *below gives the definition of work items of tasks* `Go`, `CompileQuest`, `EvaluatePhoto`, `TakePhoto`. *The second gives the definition of work items of* `SendByGPRS`. *The former group relies on the definition of predicate* $location$ *that represents the possible locations in the geographic area of interest.*

```
/* Definition of predicate location(...) identifying locations
in the geographic area of interest */
gridsize(10).
gridindex(V) :-
    gridsize(S),
    get_integer(0,V,S).
location(loc(I,J)) :- gridindex(I), gridindex(J).

/* member(ELEM,LIST) holds if ELEM is contained in LIST */
member(ELEM,[HEAD|_]) :- ELEM=HEAD.
    member(ELEM,[_|TAIL]) :- member(ELEM,TAIL).

/* Definition of predicate listelem(workitem(Task,Id,I)).
   It identifies a task Task with id Id and input I */
listelem(workitem(Task,Id,I)) :- id(Id), location(I),
    member(Task,[Go,CompileQuest,EvaluatePhoto,TakePhoto]).
listelem(workitem(SendByGPRS,Id,input)) :- id(Id).
```

□

According to the framework of Section 7.2.1, there exist two classes of fluents, domain-dependent and domain-independent. The domain-independent fluents are $enabled$ and $free$, as defined in the framework of Section 7.2.1, as well as $assigned(LWrk, Srvc)$, which is not part of the theoretical framework and has been introduced for some implementation reasons (see below in this Section). Predicate $assigned(\cdot)$ holds if a certain worklist $Lwrk$ is assigned to a service $Srvc$ as result of the execution of action $assign(Lwrk, Srvc)$. On the basis of some of these fluents we can define the four PMS actions, which are named *Primary Actions* in the terminology of the IndiGolog intepreter: $assign$, $start$, $ackTaskCompletion$ and $release$. The domain-dependent fluents can be represented in any form, relational or functional, and their successor-state axioms can be as complex as the domain needs.

Example 7.2 (cont.). *For the sake of brevity, we are showing below only the definitions of fluents* `assigned(·)` *and* $enabled(\cdot)$ *and their successor-state axioms. As far as the actions,* $assign$ *and* $release$ *can be executed in any case, whereas* $start(Task, Id, Srvc, I)$ *can be executed only if a certain work list* $LWrk$ *is assigned to* $Srvc$*, there exists an element* `workitem(Task,Id,I)` *in* $LWrk$. *Moreover,* $Task$ *has to be enabled to* $Srvc$*, which means* $Srvc$ *has previously executed action*

$readyToStart(Task, Id, Srvc)$. *The IndiGolog interpreter defines two procedures* $and(F_1, F_2)$ *and* $or(F_1, F_2)$. *The first is true if* F_1 *and* F_2 *are two formulas that hold in the current situation; the second if at least one between* F_1 *and* F_2 *holds.* F_1 *and* F_2 *are formulas that may be conjunction or disjunction of sub-formulas, which may include fluents, procedures, generic predicates, etc.*

```
/* Indicates that list LWrk of workitems has been assigned
   to service Srvc */
rel_fluent(assigned(LWrk,Srvc)) :- worklist(LWrk),
    service(Srvc).

/* assigned(LWrk,Srvc) holds after action assign(LWrk,Srvc) */
causes_val(assign(LWrk,Srvc),assigned(LWrk,Srvc),true,true).

/* assigned(LWrk,Id,Srvc) holds no longer after action
   release(LWrk,Srvc) */
causes_val(release(LWrk,Srvc),assigned(LWrk,Srvc),false,true).

/* Indicates that task Task with id Id has been begun by
   service Srvc */
rel_fluent(enabled(Task,Id,Srvc))
   :- task(Task), service(Srvc), id(Id).

/* enabled(Task,Id,Srvc) holds if the service Srvc calls
readyToStart((Task,Id,Srvc)*/
causes_val(,enabled(Task,Id,Srvc),true,true).

/* enabled(Task,Id,Srvc) holds no longer after service Srvc
calls exogenous action finishedTask(Task,Id,Srvc,V)*/
causes_val(finishedTask(Task,Id,Srvc,_),
    enabled(Task,Id,Srvc),false,true).

/* ACTIONS and PRECONDITIONS (INDEPENDENT OF THE DOMAIN) */

prim_action(assign(LWrk,Srvc)) :- worklist(LWrk),service(Srvc).
poss(assign(LWrk,Srvc), true).

prim_action(ackTaskCompletion(Task,Id,Srvc))
    :- task(Task), service(Srvc), id(Id).
poss(ackTaskCompletion(Task,Id,Srvc), neg(enabled(Task,Id,Srvc))).

prim_action(start(Task,Id,Srvc,I))
   :- listelem(workitem(Task,Id,I)), service(Srvc).
poss(start(Task,Id,Srvc,I), and(enabled(Task,Id,Srvc),
             and(assigned(LWrk,Srvc),
                 member(workitem(Task,Id,I),LWrk))
             )).

prim_action(release(LWrk,Srvc)) :- worklist(LWrk),service(Srvc).
poss(release(LWrk,Srvc), true).
```

Below we show some of the fluents that have been defined for the running example. Specifically we show fluents $at(Srvc)$ *and* $evaluationOK(Loc)$. *Careful readers may note that* at *is defined here as a functional fluent, which returns*

locations, rather than as a relational fluent. In addition, we show the abbreviation $hasConnection(Srvc)$*, which returns true if* $Srvc$ *is connected to Service* 1 *through a possible multi-hop path. Indeed, Service 1 is supposed to be deployed on the device that hosts the* MobiDis *engine. Such an abbreviation makes use of the* IndiGolog *procedure* $some(n, F(n))$ *which returns true if there exists a value* n *which makes hold formula* $F(n)$*.*

```
/* at(Srvc) indicates that service Srvc is in position P */
fun_fluent(at(Srvc)) :- service(Srvc).

causes_val(finishedTask(Task,Id,Srvc,V),at(Srvc),loc(I,J),
    and(Task=Go,V=loc(I,J))).

rel_fluent(evaluationOK(Loc)) :- location(Loc).

causes_val(finishedTask(Task,Id,Srvc,V),
    evaluationOK(loc(I,J)), true,
        and(Task=EvaluatePhoto,
        and(V=(loc(I,J),OK),
        and(photoBuild(loc(I,J),N),
            N>3)))).

proc(hasConnection(Srvc),hasConnectionHelper(Srvc,[Srvc])).

proc(hasConnectionHelper(Srvc,M),
    or(neigh(Srvc,1),
       some(n,
            and(service(n),
            and(neg(member(n,M)),
            and(neigh(n,Srvc),
            hasConnectionHelper(n,[n|M])))))))).
```

□

The realisation of the execution cycle of a work list (i.e., a list of work items) is based on procedure $isPickable(WrkList, Srvc)$. It holds if $WrkList$ is a list of proper work items and $Srvc$ is capable to perform every task defined in every work item in such a list (i.e., $Srvc$ provides all of the capabilities required).

In order to add a certain work list $WrkList$ to the process specification, designers should use procedure $manageTask(WrkList)$, which takes care of *(i)* assigning all tasks in work list $WrkList$ to one proper service, *(ii)* performing $start(\cdot)$ and $ackTaskCompletion(\cdot)$, waiting for $readyToStart(\cdot)$ $finishedTask(\cdot)$ from services, as well as *(iii)* releasing services when all tasks in the work list have been executed.

Example 7.2 (cont.). *Procedure* $manageTask(WrkList)$ *is internally composed by three sub-procedures. Firstly, it calls* $manageAssignment(WrkList)$ *that picks a certain* $Srvc$ *and turns* $assign(WrkList, Srvc)$*. Then procedure* $manageExecution(WrkList)$ *is invoked and such a procedure executes actions* $start(start(Task, Id, Srvc, I)$ *and* $ackTaskCompletion(Task, Id, Srvc)$ *one*

by one for each work item $workitem(Task, Id, I)$ in list $WrkList$. Finally, the last sub-procedure is $manageTermination(WrkList)$ which makes the picked service $Srvc$ free again by using the PMS action $realise$. It is worthy noting the use of the IndiGolog construct $atomic([a_1; \ldots; a_n])$ to provide an atomic execution of a action sequence $a_1, \ldots, a_n$. Here atomicity is intended in the sense that all of these actions are performed sequentially and any other procedure is stuck till the whole sequence execution. For instance, in procedure $manageAssignment$ the atomic constructs is used in order to prevent the same service to be picked by different executions of procedure $manageAssignment$. Otherwise, this would cause obvious inconsistences.

Procedure $isExecutable$ uses the IndiGolog construct $findall(elem, formula, set)$, which works as follows: it takes all instances of $elem$ that makes $formula$ true and puts all of them in set set. $elem$ and $formula$ are unified by the same term name; that means $formula$ has to have a non-ground term named $elem$. Being that said, in procedure $isExecutable(Task, Srvc)$ term A denotes the set of all capabilities required by task $Task$, whereas C denotes all capabilities provided by service $Srvc$. When can $Srvc$ execute $Task$? If the set A of the capabilities required by $Task$ is a sub set of C, the capabilities provided by $Srvc$.

```
proc(isPickable(WrkList,Srvc),
    or(WrkList=[],
        and(free(Srvc),
        and(WrkList=[A|TAIL],
        and(listelem(A),
        and(A=workitem(Task,_Id,_I),
        and(isExecutable(Task,Srvc),
        isPickable(TAIL,Srvc))))))
    )
).

proc(isExecutable(Task,Srvc),
    and(findall(Capability,required(Task,Capability),A),
    and(findall(Capability,provide(Srvc,Capability),C),
        subset(A,C)))).

/* PROCEDURES FOR HANDLING THE TASK LIFE CYCLES */

proc(manageAssignment(WrkList),
    [atomic([pi(Srvc,[?(isPickable(WrkList,Srvc)),
        assign(WrkList,Srvc)])])]).

proc(manageExecution(WrkList),
    pi(Srvc,[?(assigned(WrkList,Srvc)=true),
        manageExecutionHelper(WrkList,Srvc)])).

proc(manageExecutionHelper([],Srvc),[]).

proc(manageExecutionHelper([workitem(Task,Id,I)|TAIL],Srvc),
```

```
    [start(Task,Id,Srvc,I), ackTaskCompletion(Task,Id,Srvc),
        manageExecutionHelper(TAIL,Srvc)]).

proc(manageTermination(WrkList),
    [atomic([pi(n,[?(assigned(WrkList,n)=true),
        release(X,n)])])]).

proc(manageTask(WrkList),
    [manageAssignment(WrkList),
     manageExecution(WrkList),
     manageTermination(WrkList)]).
```

□

Finally, if the framework is properly configured, the program that codes a process results to be quite simple. Specifically, for the running example is the following:

```
proc(branch(Loc),
    while(neg(evaluationOk(Loc)),
      [
        manageTask([workitem(CompileQuest,id_1,Loc)]),
        manageTask([workitem(Go,id_1,Loc),
                    workitem(TakePhoto,id_2,Loc)]),
        manageTask([workitem(EvaluatePhoto,id_1,Loc)]),
      ]
    )
).

proc(process,
[rrobin([branch(loc(2,2),branch(loc(3,5)),branch(loc(4,4)))]),
 manageTask([workitem(SendByGPRS,id_29,input)])
] ).
```

Chapter 8

Conclusions

The work developed in this thesis is involved in European-funded project WORKPAD. This project aims at designing and developing an innovative service-oriented architecture for supporting collaborative work of human operator and non-human service in emergency management scenarios.

This thesis has shown an innovative methodology used for designing the WORKPAD interactive system. The process of designing interactive systems didn't involve only the interfaces or the immediate interaction; rather, it related to the complete environment (e.g. the technology, working environment, stakeholders, goals, etc.) surrounding that system; and this process produced a set of resulting artefacts (e.g. documentation, manuals, tutorial, etc.) along with the working system to make the interaction more effective. The outcomes and results obtained by this Thesis have been published in the following conferences/journals:

- **Springer's International Journal on Knowledge, Technology and Policy.** S.R. Humayoun, T. Catarci, M. de Leoni, A. Marrella, M. Mecella, M. Bortenschlager, R. Steinmann. *"Designing Mobile Systems in Highly Dynamic Scenarios. The WORKPAD Methodology."* Volume 22, Number 1, March, 2009.

- **IEEE Internet Computing Journal.** T. Catarci, M. de Leoni, A. Marrella, M. Mecella, G. Vetere, S. Dustdar et al. *"Pervasive and Peer-to-Peer Software Environments for Supporting Disaster Responses"*. Special Issue on Crisis Management, January, 2008.

- **Session on Flexible Service and Data Management Platforms for Crisis Response Session at the 4th International Conference on Information Systems for Crisis Response and Management (ISCRAM).** M. de Leoni, A. Marrella, M. Mecella, F. De Rosa, M. Mecella, A. Poggi, A. Krek, F. Manti. "*Emergency Management: from User Requirements to a Flexible P2P Architecture*". May, 2007.

Furthermore, the psychological approach we used to design the user interface of PMS has been appreciated and published in :

- **13th International Conference on Human-Computer Interaction (HCI International 2009).** S.R. Humayoun, T. Catarci, M. de Leoni, A. Marrella, M. Mecella, M. Bortenschlager, R. Steinmann. *"The WORKPAD User Interface and Methodology: Developing Smart and Effective Mobile Applications for Emergency Operators"*, 19-24 July, 2009.

Finally, this thesis has collaborated in the realization of the PMS IndiGolog, that is a new approach to the process management. Also in this case, this relevant result has been published in :

- **2nd IEEE International Workshop on Interdisciplinary Aspects of Coordination Applied to Pervasive Environments: Models and Applications (COMA 2008) at WETICE 08.** M. de Leoni, A. Marrella, M. Mecella, S. Valentini, S. Sardina. *"Coordinating Mobile Actors in Pervasive and Mobile Scenarios: An AI-based Approach"*. June, 2008.

By analyzing this Thesis, it results that the followed User-Centered approach has led to a successful system. This same approach is actually used in many other European Project; as an instance, the recently funded SM4ALL Project[1] uses a User-Centered methodology to design the system, since its focus is about disabled people.

A possible future work could concerned an improvement the of the User-Centered approach, probably too broad in some points. Moreover, a further improvement of the PMS IndiGolog code could be interesting for the research point of view, in the field of situation calculus. However, the big challenge that this work wants to launch is about the design and implementation of a vocal module working on PDA during an emergency.

[1] http://www.sm4all-project.eu/

Bibliography

[1] *Human-Centered Design Processes for Interactive Systems; International Standardization Organization: Human-Centered Design Processes For Interactive Systems*, 1999.

[2] *GIS for Emergency Management*, 2005. http://www.esri.com/library/whitepapers/pdfs/emermgmt.pdf.

[3] G. Abowd A. Dix, J. Finlay and R. Beale. *Human Computer Interaction (3rd edition)*. Prentice Hall, 2003.

[4] F. De Rosa M. Mecella A. Poggi A. Krek A. Marrella, M. Mecella and F. Manti. Emergency Management: from User Requirements to a Flexible P2P Architecture. 2009.

[5] D.I. Smith A. Zerger. Impediments to using GIS for Real-time Disaster Decision Support. *Computers,Environment and Urban Systems*, 27(2):123–141, 2003.

[6] J. Annett and K.D. Duncan. Task Analysis and Training Design. *Occupational Psychology*, 1967.

[7] J. Baier and S. McIlraith. On Planning with Programs that Sense. In *KR'06: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning*, pages 492–502, Lake District, UK, June 2006. AAAI Press.

[8] Orchestra Executive Board. *Towards an open disaster Risk management service architecture*, 2005. http://www.eu-orchestra.org/.

[9] B. Boehm. A Spiral Model of Software Development and Enhancement. *IEEE*, pages 61–72, 1988.

[10] F. Brooks. *The Mythical Man-Month*. Addison-Wesley, 1975.

[11] AMIRA Consortium. *D2.1.0 The Socio-Economic Study of User Needs*, 2004. http://www.amira.no/.

[12] LIAISON Consortium. *Mission Requirements Document D056, Issue 3.E*, 2006. http://www.newapplication.it/liaison/.

[13] LIAISON Consortium. *Service Definition Pre-op D055, Issue 3.D*, 2006. http://www.newapplication.it/liaison/.

[14] OASIS Consortium. *OASIS User Requirements synthesis, Version 3.1*, 2005. http://www.oasis-fp6.org/.

[15] ORCHESTRA Consortium. *Reference Model for the ORCHESTRA Architecture, Annex A2 - Requirements for the ORCHESTRA Architecture and ORCHESTRA Service Networks*, 2007. http://www.eu-orchestra.org/.

[16] WIN Consortium. *Deliverable D3201 User Requirements Specifications, Revision 2.00*, 2006. http://www.win-eu.org/.

[17] European Council. Council decision of 23rd October 2001, establishing a Community mechanism to facilitate reinforced cooperation in civil protection assistance interventions, 2001.

[18] J. Crinnion. *Evolutionary Systems Development, a practical guide to the use of prototyping within a structured systems methodology*. Plenum Press, 1991.

[19] S. Draper D. Norman. *User-Centered System Design: New Perspectives on Human-Computer Interaction*. LAWRENCE ERLBAUM ASSOCIATES, 1986.

[20] G. De Giacomo and H.J. Levesque. An incremental interpreter for high-level programs with sensing. In Hector J. Levesque and Fiora Pirri, editors, *Logical Foundations for Cognitive Agents: Contributions in Honor of Ray Reiter*, pages 86–102. Springer, Berlin, 1999.

[21] Massimiliano de Leoni. Adaptive Process Management in Highly Dynamic and Pervasive Scenarios. Master's thesis, University of Rome "La Sapienza", 2009.

[22] S. Denning. *The Springboard - How Storytelling Ignites Action in Knowledge-Era Organizations*. Butterworth-Heinemann, 2001.

[23] K. Eason. *Information Technology and Organisational Change*. Taylor and Francis, 1988.

[24] A. Malizia F. De Rosa and M. Mecella. Disconnection Prediction in Mobile Ad hoc Networks for Supporting Cooperative Work. *IEEE Pervasive Computing*, 4, 2005.

[25] H. J. Levesque G. De Giacomo and S. Sardina. Incremental execution of guarded theories. *ACM Transactions on Computational Logic (TOCL)*, October 2001.

[26] H.J. Levesque G. De Giacomo, Y. Lespérance and S. Sardina. *IndiGolog: A High-Level Programming Language for Embedded Reasoning Agents*. 2009.

[27] R. Reiter G. De Giacomo and M. Soutchanski. Execution Monitoring of High-Level Robot Programs. In *KR'98: Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning*, pages 453–465, 1998.

[28] E. Galanti. *Guida del Dipartimento della Protezione Civile*, 2004. http://www.casaleinforma.it/pcivile/scarica/04augustus.pdf.

[29] G. Booch I. Jacobson and J. Rumbaugh. *The Unified Software Development Process*. Addison Wesley, 1999.

[30] IBM. *Web Service Notification Specification*, 2005. http://www-128.ibm.com/developerworks/library/specification/ws-notification/.

[31] Y. Rogers J. Preece and H. Sharp. *Interaction Design*. Apogeo, 2002.

[32] Bartek Kiepuszewski, Arthur H. M. ter Hofstede, and Christoph Bussler. On Structured Workflow Modelling. In *CAiSE '00: Proceedings of the 12th International Conference on Advanced Information Systems Engineering*, pages 431–445, London, UK, 2000. Springer-Verlag.

[33] B. Kirwan and L.K. Ainsworth. *A Guide to Task Analysis*. Taylor and Francis, 1992.

[34] Robert A. Kowalski. Using meta-logic to reconcile reactive with rational agents. *Meta-logics and logic programming*, pages 227–242, 1995.

[35] R.A. Krueger and M.A. Casey. *Focus Groups: A Practical Guide for Applied Research*. SAGE Publications Ltd, 2000.

[36] Yves Lespérance and Ho-Kong Ng. Integrating Planning into Reactive High-Level Robot Programs. In *Proceedings of the Second International Cognitive Robotics Workshop (in conjunction with ECAI 2000)*, pages 49–54, August 2000.

[37] M. Mecella M. de Leoni and G. De Giacomo. Highly dynamic adaptation in process management systems through execution monitoring. *BPM*, 2007.

[38] M. Mecella S. Valentini S. Sardina M. de Leoni, A. Marrella. Coordinating Mobile Actors in Pervasive and Mobile Scenarios: An AI-based Approach. 2008.

[39] L.A. Macaulay. *Requirements Engineering*. Springer Verlag, 1996.

[40] J. Martin. *Rapid Application Development*. Macmillan Coll Div, 1991.

[41] Stefano Menotti. SPIDE A Smart Process IDE for Emergency Operators. Master's thesis, Faculty of Computer Engineering - SAPIENZA Università di Roma, 2008. Supervisor: Dr. Massimo Mecella. In English.

[42] J. Nielsen. *Usability Engineering*. Academic Press, 1993.

[43] N. Rackham. *SPIN Selling*. McGraw Hill, 1996.

[44] R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, September 2001.

[45] W. Royce. Managing the Development of Large Software Systems. *IEEE WESCON*, pages 328–338, 1970.

[46] E. Vogel S. Luck. The Capacity of Visual Working Memory for Features and Conjunctions. *Nature*, 390:279Ű281, 2002.

[47] B. Shneiderman. *Designing the User Interface : Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1987.

[48] J. Sodhi and P. Sodhi. *Managing IT System Requirements*. Management Concepts, 2003.

[49] M. de Leoni A. Marrella M. Mecella M. Bortenschlager S.R. Humayoun, T. Catarci and R. Steinmann. Designing Mobile Systems in Highly Dynamic Scenarios. The WORKPAD Methodology. 2009.

[50] M. de Leoni A. Marrella M. Mecella M. Bortenschlager S.R. Humayoun, T. Catarci and R. Steinmann. The WORKPAD User Interface and Methodology: Developing Smart and Effective Mobile Applications for Emergency Operators. 2009.

[51] R. Sternberg. *Cognitive Psychology*. Wadsworth Publishing, 2002.

[52] A.G. Sutcliffe. Scenario-Based Requirements Engineering. *11th IEEE International Conference on Requirements Engineering*, 2003.

[53] A. Marrella M. Mecella G. Vetere B. Salvatore et al. T. Catarci, M. de Leoni. Pervasive Software Environments for Supporting Disaster Responses. *IEEE Computer Society*, January 2008.

[54] European Union. *Mechanisms for Assistance Interventions*, 2007. http://europa.eu/scadplus/leg/en/lvb/l28003.htm.

[55] European Union. *The Common Emergency Communication and Information System CECIS*, 2007. http://ec.europa.eu/environment/civil/cecis.htm.

[56] European Union. *The Monitoring and Information Centre MIC*, 2007. http://ec.europa.eu/environment/civil/prote/mic.htm.

[57] W. van der Aalst and K. van Hee. *Workflow Management. Models, Methods, and Systems*. MIT Press, 2004.

[58] R.A. Virzi. Refining the test phase of usability evaluation: How many subjects is enough? *Human Factors*, 34:457–468, 1992.

[59] K. Wiegers. Karl Wiegers Describes 10 Requirements Traps to Avoid. *Software Testing and Quality Engineering Journal*, 2000.

[60] K. Wiegers. *More About Software Requirements: Thorny Issues and Practical Advice*. Microsoft Press, 2006.

[61] Jan Wielemaker. An Overview of the SWI-Prolog Programming Environment. In *WLPE: Proceedings of the 13th International Workshop on Logic Programming Environments*, volume CW371 of *Report*, pages 1–16, 2003.