Gestione parcheggi

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Analisi e specifica dei requisiti 4

2.3 Use case 6

2.4 Pianificazione 6

2.5 Analisi dei mezzi 6

2.5.1 Software 6

2.5.2 Hardware 6

3 Progettazione 7

3.1 Design dell’architettura del sistema 7

3.2 Design dei dati e database 7

3.3 Design delle interfacce 7

3.4 Design procedurale 7

4 Implementazione 8

5 Test 8

5.1 Protocollo di test 8

5.2 Risultati test 9

5.3 Mancanze/limitazioni conosciute 9

6 Consuntivo 9

7 Conclusioni 9

7.1 Sviluppi futuri 9

7.2 Considerazioni personali 9

8 Bibliografia 9

8.1 Bibliografia per articoli di riviste: 9

8.2 Bibliografia per libri 9

8.3 Sitografia 9

9 Allegati 10

# Introduzione

## Informazioni sul progetto

Allievo: Paolo Weishaupt

Classe: Informatica 4AC presso la Scuola d’Arti e Mestieri di Trevano

Docente responsabile: Guido Montalbetti

Data inizio: 03/09/2019

Data fine: 20/12/2019

## Abstract

E’ una breve e accurata rappresentazione dei contenuti di un documento, senza notazioni critiche o valutazioni. Lo scopo di un abstract efficace dovrebbe essere quello di far conoscere all’utente il contenuto di base di un documento e metterlo nella condizione di decidere se risponde ai suoi interessi e se è opportuno il ricorso al documento originale.

Può contenere alcuni o tutti gli elementi seguenti:

* **Background/Situazione iniziale**
* **Descrizione del problema e motivazione**: Che problema ho cercato di risolvere? Questa sezione dovrebbe includere l'importanza del vostro lavoro, la difficoltà dell'area e l'effetto che potrebbe avere se portato a termine con successo.
* **Approccio/Metodi**: Come ho ottenuto dei progressi? Come ho risolto il problema (tecniche…)? Quale è stata l’entità del mio lavoro? Che fattori importanti controllo, ignoro o misuro?
* **Risultati**: Quale è la risposta? Quali sono i risultati? Quanto è più veloce, più sicuro, più economico o in qualche altro aspetto migliore di altri prodotti/soluzioni?

Esempio di abstract:

*As the size and complexity of today’s most modern computer chips increase, new techniques must be developed to effectively design and create Very Large Scale Integration chips quickly. For this project, a new type of hardware compiler is created. This hardware compiler will read a C++ program, and physically design a suitable microprocessor intended for running that specific program. With this new and powerful compiler, it is possible to design anything from a small adder, to a microprocessor with millions of transistors. Designing new computer chips, such as the Pentium 4, can require dozens of engineers and months of time. With the help of this compiler, a single person could design such a large-scale microprocessor in just weeks.*

## Scopo

Lo scopo del progetto (scopi didattici/scopi operativi). Dovrebbe descrivere il mandato, ma non vanno ricopiate le informazioni del quaderno dei compiti (che va invece allegato).

# Analisi

## Analisi del dominio

Questo capitolo dovrebbe descrivere il contesto in cui il prodotto verrà utilizzato, da questa analisi dovrebbero scaturire le risposte a quesiti quali ad esempio:

* Background/Situazione iniziale
* Quale è e come è organizzato il contesto in cui il prodotto dovrà funzionare?
* Come viene risolto attualmente il problema? Esiste già un prodotto simile?
* Chi sono gli utenti? Che bisogni hanno? Come e dove lavorano?
* Che competenze/conoscenze/cultura posseggono gli utenti in relazione con il problema?
* Esistono convenzioni/standard applicati nel dominio?
* Che conoscenze teoriche bisogna avere/acquisire per poter operare efficacemente nel dominio?
* …

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Creazione applicativo web |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il progetto vuole creare un sito Web dove poter gestire i parcheggi disponibili della scuola |
| **Sotto requisiti** | |
| **001** | Le persone potranno ricercare i posteggi disponibili |
| **002** | Le persone potranno riservare i posteggi |
| **003** | Le persone potranno stampare la loro riservazione |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Creazione maschera admin |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il sito dovrà avere una parte del menù riservata agli amministratori |
| **Sotto requisiti** | |
| **001** | Gli amministratori potranno inserire i parcheggi disponibili |
| **002** | Gli amministratori potranno gestire le persone registrate al sito |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Funzionalità principali per gli utenti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il sito dovrà avere due sezioni principali: Offerta e Ricerca |
| **Sotto requisiti** | |
| **001** | Nella sezione Offerta l’utente dovrà essere in grado di mettere a disposizione il suo parcheggio. |
| **002** | Nella sezione Offerta le persone potranno inserire le date della disponibilità del loro posto auto (mattina, pomeriggio o tutto il giorno) e in più indicare il loro numero di targa oltre ai loro dati personali di base |
| **003** | Nella sezione “Ricerca” si potranno ricercare i parcheggi a disposizione inserendo come parametro il giorno preciso o anche un periodo, p.es. dal 2.12.2019 al 5.12.2019 |
| **004** | Sulla lista dei risultati, fornita dalla ricerca, si potranno ordinare e filtrare i dati |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Creazione maschera di login |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Le persone dovranno registrarsi al sito ed inserire i propri dati principali per poter riservare il posto auto |
| **Sotto requisiti** | |
| **001** | I campi minimi da compilare saranno: nome, cognome, email e un recapito telefonico |
| **002** | L’applicativo dovrà prevedere un meccanismo di conferma della registrazione per l’identificazione della persona, p.es. l’invio di un link all’indirizzo di posta elettronica della persona, impostato in fase di registrazione |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Specifiche per la riservazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Di seguito le specifiche da seguire per la riservazione di un posteggio |
| **Sotto requisiti** | |
| **001** | Il costo del posto auto è di 10 CHF al giorno da conteggiare e fatturare mensilmente (mezza giornata sono 5 CHF) |
| **002** | Al fronte di una riservazione, le persone riceveranno una conferma tramite email |
| **003** | Prevedere un meccanismo di concorrenzialità per evitare che una persona in procinto di riservare il parcheggio, si veda portar la sua riservazione da un’altra persona |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Gestione stampe |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Di seguito le specifiche da seguire per la gestione delle stampe |
| **Sotto requisiti** | |
| **001** | Gli amministratori potranno stampare la situazione dei parcheggi per un determinato periodo (a disposizione, riservati) |
| **002** | Gli amministratori potranno stampare la fattura per una persona singola o per tutte le persone, scegliendo un periodo da loro impostato, generalmente una stampa mensile. Il saldo della fattura è di 30 giorni netto |
| **003** | Le persone che avranno riservato il parcheggio potranno stampare la loro riservazione |
| **004** | Gli amministratori potranno stampare i richiami per le persone che non hanno ancora saldato la loro riservazione. A metà di ogni mese, giorno feriale, verranno stampati i richiami delle fatture. Prevedere un meccanismo automatico con notifica all’utente prima della stampa |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | Gestione fatturazione |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Di seguito le specifiche da seguire per la gestione delle fatturazioni |
| **Sotto requisiti** | |
| **001** | Se dopo il primo e unico richiamo la persona non avesse ancora saldato la fattura, la persona sarà “bloccata” e non avrà più la possibilità di riservare i parcheggi sino al saldo della fattura. In questo caso la persona interessata sarà avvisata via email del suo blocco e del richiamo non ancora saldato |

|  |  |
| --- | --- |
| **ID: REQ-008** | |
| **Nome** | Convalida dati |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’applicativo avrà una convalida dei dati |
| **Sotto requisiti** | |
| **001** | La convalida dei dati immessi è una funzionalità dell’applicativo, p.es. l’email e il numero di telefono devono avere un formato corretto |

## Use case

I casi d’uso rappresentano l’interazione tra i vari attori e le funzionalità del prodotto.

## Pianificazione

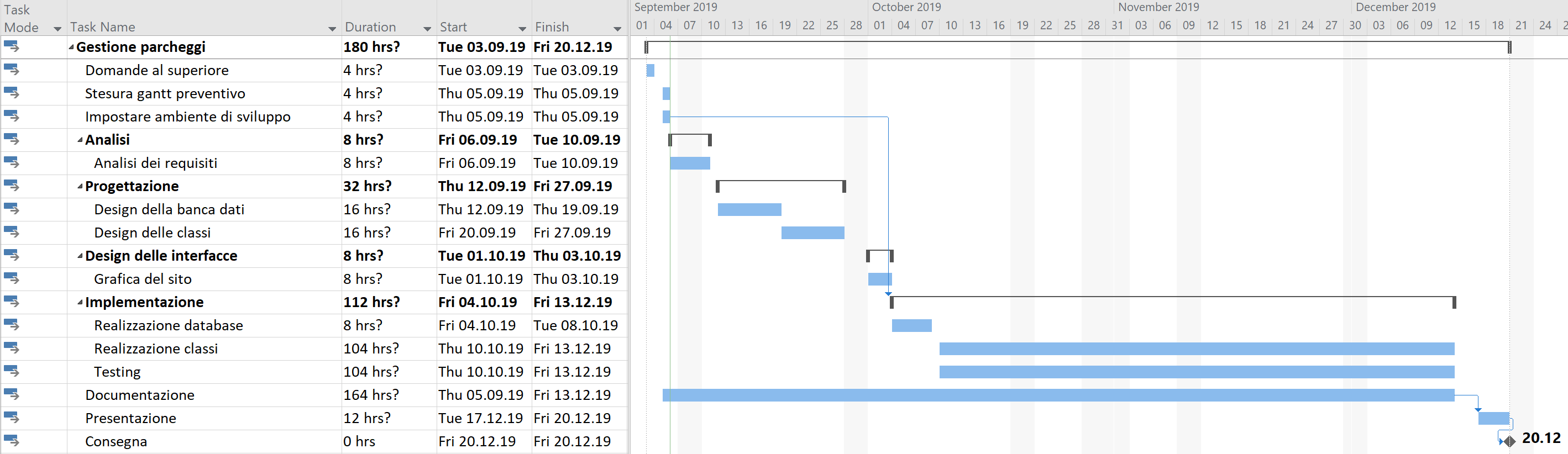


Figura 1 Gantt preventivo

## Analisi dei mezzi

### Software

I software utilizzati per la realizzazione di questo progetto sono:

* Microsoft Word 2016 - usato fino al 13/09/2019
* WPS Office Free 11.1.0.8865
* Visual Studio Code- usato fino al 15/10/2019
* PhpStorm-192.6817.20
* Project
* Google Chrome
* VMWare Workstation - usato fino al 20/09/2019
* XAMPP 7.3.0-0 - usato fino al 13/09/2019
* MySQL Workbench 8.0 CE
* draw.io online
* SourceTree - usato fino al 13/09/2019

### Hardware

Computer Paolo Weishaupt:

* Modello: Huawei MateBook X Pro
* Processore: Intel Core i7-8550U
* RAM: 8GB
* Display 13.9” 3000x2000 px
* SSD: 512 GB
* OS: Windows 10 Home - usato fino al 10/09/2019
* OS: Deepin 15.11 basato su Debian 9 Stretch

# Progettazione

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all’esecutore di evitare fraintendimenti e imprecisioni nell’implementazione del prodotto.

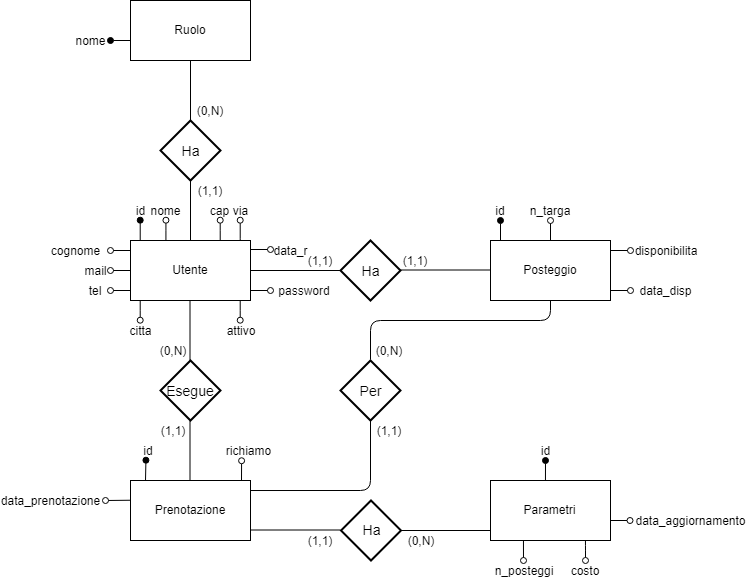
## Design dell’architettura del sistema

Descrive:

* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap

## Design dei dati e database

### Diagramma E-R



### Schema procedurale

Parametri(id, n\_posteggi, costo, data\_aggiornamento)

Ruolo(nome)

Utente(id, ruolo(FK), nome, cognome, mail, via, cap, citta, tel, data\_r, attivo, password, id\_posteggio(FK))

Posteggio(id, disponibilita, data\_disp, n\_targa)

Prenotazione(id, id\_utente(FK), id\_posteggio(FK), id\_parametri(FK), richiamo, data\_prenotazione)

### Descrizione database

Il database è composto da 5 entità.

L’entità PARAMETRI serve per contenere il numero di parcheggi disponibili, il costo di affitto di un parcheggio e la data dell’ultimo aggiornamento del costo.

L’entità RUOLO contiene solo un campo ed è il nome del ruolo. Questo perché se in futuro si dovesse avere il bisogno di cambiare i nomi dei ruoli non si dovrà cambiare manualmente per tutti gli utenti.

L’entità UTENTE rappresenta gli utenti del db e contiene le loro informazioni. L’attributo attivo serve per la verifica dell’e-mail perché finché non viene verificata il suo valore sarà su false e l’account sarà disabilitato. L’attributo potrà inoltre essere settato a false se l’utente dovesse avere dei richiami su una fattura.

L’entità POSTEGGIO rappresenta un posteggio e contiene i dati necessari al suo riconoscimento. Gli attributi disponibilita, data\_disp e n\_targa verranno utilizzati se il parcheggio dovrà essere messo in offerta.

L’entità PRENOTAZIONE serve per gestire le prenotazioni degli utenti di un parcheggio. L’attributo richiamo ha di default il valore false. Se la fattura riguardante quella prenotazione non dovesse essere saldata verrà settato a true e al prossimo richiamo l’utente verrà disabilitato.

## Design delle interfacce

Queste sono le interfacce che ho previsto di avere nel mio programma. Il design finale potrà differire da quello inizialmente progettato.

### Pagina di login

### 3.3.2

## Design procedurale

Descrive i concetti dettagliati dell’architettura/sviluppo utilizzando ad esempio:

* Diagrammi di flusso e Nassi.
* Tabelle.
* Classi e metodi.
* Tabelle di routing
* Diritti di accesso a condivisioni …

Questi documenti permetteranno di rappresentare i dettagli procedurali per la realizzazione del prodotto.

# Implementazione

In questo capitolo dovrà essere mostrato come è stato realizzato il lavoro. Questa parte può differenziarsi dalla progettazione in quanto il risultato ottenuto non per forza può essere come era stato progettato.

Sulla base di queste informazioni il lavoro svolto dovrà essere riproducibile.

In questa parte è richiesto l’inserimento di codice sorgente/print screen di maschere solamente per quei passaggi particolarmente significativi e/o critici.

Inoltre dovranno essere descritte eventuali varianti di soluzione o scelte di prodotti con motivazione delle scelte.

Non deve apparire nessuna forma di guida d’uso di librerie o di componenti utilizzati. Eventualmente questa va allegata.

Per eventuali dettagli si possono inserire riferimenti ai diari.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-012 | **Nome:** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione:** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti:** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura:** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi:** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap 1.7) (ad esempio Gannt consuntivo).

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo,

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o Qdc
* Prodotto
* …