INLIGTINGSTELSELINGENEURSWESE II
INFORMATION SYSTEMS ENGINEERING II

STUDIEGIDS VIR / STUDY GUIDE FOR

ITRI624VBC

SKOOL VIR INLIGTINGSTEGNOLOGIE
SCHOOL OF INFORMATION TECHNOLOGY

VAALDRIEHOEKKAMPUS / VAAL TRIANGLE CAMPUS

Studiegids saamgestel deur / Study guide compiled by:

Prof Magda Huisman

Studiegids aangepas deur / Study guide adopted by:

Dr S Gilliland

(Skool vir Inligtingstegnologie / School of Information Technology)

Taalsorg nn.

Bladuitleg deur Elsabe Strydom, **graphikos.**

Hantering van drukwerk en verspreiding deur Departement Logistiek (Verspreidingsentrum).

Gedruk deur Ivyline Technologies (018) 293 0715/6.

INHOUDSOPGAWE

MODULE CONTENTS

MODULE-INLIGTING

Modulekode: ITRI624

Beskrywende naam: Inligtingstelsel-ingenieurswese II

Krediete 12 (120 uur)

Kontakperiodes Twee teorie

TYDVERDELING

Aktiwiteit	**Weke**	**Ure/week**	**Totaal**
Klas (kontaktyd)	12	2	24
Projek	12	3.5	42
Selfstudie en voorbereiding	12	2	24
Semestertoets en voorbereiding	1	10	10
Eksamen en voorbereiding	1	20	20
Totaal			120

ALGEMEEN:

Veronderstelde leer: ITRW213 en ITRW225

Dosent: Dr Daan de Villiers, Dr S Gilliland

Gebou 8, Kamer 44

Telefoon: 016 910 3265/1

E-pos: Sonja.Gilliland@nwu.ac.za

Spreekure: Soos op kantoordeur aangebring

STUDIEMATERIAAL

Voorgeskrewe handboek

David Avison and Guy Fitzgerald: *Information Systems Development: Methodologies, Techniques and Tools*. Vierde uitgawe. Uitgegee deur Mc Graw Hill.

ISBN 0-07-711417-5

KLASBYWONING

Klasbywoning vorm ‘n sentrale deel van hierdie module. Dit is noodsaaklik dat alle klasse bygewoon word.

MODULE INFORMATION

Module Code:	ITRI624
Descriptive Name:	Information System Engineering II
Credits:	120 hours
Contact sessions:	Two theoretical

DIVISION OF TIME

Activities	Weeks	Hours/week	Total
Class contact time	12	2	24
Project	12	3.5	42
Self study and preparation	12	2	24
Semester test and preparation	1	10	10
Exams and preparation	1	20	20
Total			120

GENERAL:

Assumed study:	ITRW213 and ITRW225
Lecturer:	Dr Daan de Villiers, Dr S Gilliland
	Building 8 Room 44
	Telephone: 016 910 3265/1 Email:
	sonja.gilliland@nwu.ac.za
Consulting-hours:	See office door

STUDY MATERIAL

Prescribed handbook

Information Systems Development: Methodologies, Techniques and Tools – David Avison and Guy Fitzgerald. Fourth Edition. Published by Mc Graw Hill.

ISBN 0-07-711417-5

CLASS ATTENDANCE

Class attendance forms an integral part of this module. It is essential that all classes be attended.

WERKSWYSE

Selfwerksaamheid is van uiterste belang in hierdie kursus. Die studiegids verskaf 'n raamwerk van die semester se werk. Studente moet voor 'n klas die werk aandagtig deurwerk en die opdragte doen. Die dosent tree op as fasiliteerder in die kursus. Tydens klasperiodes word studente se probleme met die werk en die opdragte bespreek. Studente is welkom om navrae deur middel van e-pos aan die dosent te rig.

EVALUERING

U word op verskeie wyses getoets deur die semester om te bepaal of u die uitkomstes bereik het.

Huiswerkopdragte

- Studente ontvang gereeld opdragte.
- Leer is ’n kumulatiewe proses; nuwe kennis bou op voorkennis. Die opdragte is dus noodsaaklik om te kontroleer dat die uitkomste vir elke deel van die werk bemeester is.
- Opdragte word aan die begin van die volgende klasgeleentheid ingedien. Geen laat opdragte word aanvaar nie.

Semestertoets

- Toetse sal op geskeduleerde tye gedurende die semester geskryf word.

Praktiese projek

- Studente ontvang individueel ‘n praktiese projek wat gedurende die semester voltooi moet word.
- Tydens hierdie projek word die teoretiese beginsels wat behandel is, op ‘n realistiese gevallestudie toegepas. Verder word van die studente verwag om self inligting in te win oor nuwe onderwerpe wat in die projek na vore kom.

Deelnamepunt

Die deelnamepunt word ongeveer soos volg bereken:

Opdragte	20
Praktiese projek	30
Toetse	50

METHOD OF WORK

Self-activity is of utmost importance in this course. The study guide gives you a framework of the work in this semester. It is expected of you to do the work beforehand and to do the assignments given to you. The lecturer acts as a facilitator during the course; during class periodes the students' problems with the assignments will be dealt with. Students are welcome to state their questions to the lecturer by e-mail.

EVALUATION

You will be tested in various ways throughout the semester to establish whether you have been successful in attaining the outcomes.

Homework assignments

- Students will regularly receive assignments.
- Studying is a cumulative process and knowledge is built on previous knowledge. It is therefore necessary to control the assignments to establish if the outcomes of each part of the work have been mastered.
- Assignments should be submitted at the beginning of the following class period. No late assignments will be accepted.

Tests

- Tests will be written on sheduled times during the semester.

Practical project

- Students will each receive a practical project which has to be completed during the semester.
- During this project the theoretical principles that were dealt with, will be applied on a realistic case study. Furthermore, it will be expected from students to obtain information on new topics that came forward in the project.

Participation mark

The participation mark will be estimated roughly as follows:

Assignments	20
Practical project	30
Tests	50

Final mark

The final mark will be estimated roughly as follows:

Participation mark	50
Exam	50

Finale punt

Die finale punt word ongeveer soos volg bereken:

Deelnamepunt	50
Eksamen	50

WENKE OM HIERDIE MODULE TE SLAAG

- Wees goed voorberei vir elke periode. Doen ALLE opdragte vooraf en noem die probleme wat u daarmee ondervind aan die dosent tydens die periode.
- Neem aktief deel aan die besprekings. Die dosent gee nie formeel 'n lesing nie, maar lei besprekings oor onderwerpe wat in die leereenheid na vore kom.
- Werk van die begin van die semester af aan die praktiese projek. Die omvang van die projek is baie groot, en u sal nie suksesvol wees as u eers 'n week of twee voor die doeldatum begin werk nie.

WERKSPROGRAM

WEEK	Leereenheid	ONDERWERP
1	1	Inleiding tot inligtingstelselontwikkeling
2	2	Lewensiklusbenadering
3	3	Prosesgeoriënteerde stelselontwikkelingmetodologieë
4	4	Data (gemengde) -georiënteerde stelselontwikkelingmetodologieë
5	5	Objekgeoriënteerde stelselontwikkelingmetodologieë
6	6	RAD-georiënteerde stelselontwikkelingmetodologieë
7	7	Mensgeoriënteerde stelselontwikkelingmetodologieë
8	8	Organisasiegeoriënteerde stelselontwikkelingmetodologieë
9	9	Raamwerke
10	10	Inligtingstelselontwikkeling sonder metodologieë
11	11	Relevante kwessies rakende stelselontwikkelingmetodologieë
12	12	Vergelyking van stelselontwikkelingmetodologieë
13	-	Indiening van praktiese projek

HINTS ON HOW TO PASS THIS MODULE

- Be well prepared for every period. Do ALL the assignments beforehand and stipulate the problems that you have to the lecturer during this period.
- Actively take part in discussions. The lecturer does not give a formal lecture, but leads discussions on subjects that were dealt with in the study unit.
- Start working on the practical project from the beginning of the semester. The range of the project is quite large and you will not be successful if you only start working a week or two before the due date.

WORK PROGRAMME

WEEK	Study unit	SUBJECT
1	1	Introduction to information system development
2	2	Life cycle approach
3	3	Process oriented system development methodologies
4	4	Data (blended) oriented system development methodologies
5	5	Object oriented system development methodologies
6	6	RAD oriented system development methodologies
7	7	People oriented system development methodologies
8	8	Organisational oriented system development methodologies
9	9	Frameworks
10	10	Information system development without methodologies
11	11	Relevant matters regarding system development methodologies
12	12	Comparison of system development methodologies
13	-	Submit practical project

MODULE-UITKOMSTE

Doel:

Verskaffing van ‘n goeie kennis en praktiese ervaring van verskillende stelselontwikkelings-metodologieë aan die leerder

Kennis:

Aan die einde van die studie sal die studente goeie kennis dra van verskillende stelselontwikkelingsmetodologieë. Dit sluit in stelselontwikkelingsmetodes, onderliggende benaderings waarop stelselontwikkelingsmetodes gebaseer is, ontwikkelingsprosesmodelle wat gevolg word in stelselontwikkelingsmetodes, ontwikkelingstegnieke en hulpmiddels wat gebruik word in stelselontwikkelingsmetodes.

Studente behoort na die voltooiing van hierdie module

- Inligtingstelsel-ingenieurswese te kan definieer en verduidelik.
- ‘n Stelselontwikkelingsmetodologie te kan definieer en verduidelik.
- Die aanvaarding van stelselontwikkelingsmetodologie in praktyk te kan verduidelik.
- STRADIS (Structured analysis, design, and implementation of information systems) te verstaan en te kan toepas.
- IE (Information engineering) te verstaan en te kan toepas.
- RUP (Rational Unified Process) te verstaan en te kan toepas.
- XP (Extreme Programming) te verstaan en te kan toepas.
- SSM (Soft Systems Methodology) te verstaan en te kan toepas.
- ETHICS (Effective technical and human implementation of computer-based systems) te verstaan en te kan toepas.
- MULTIVIEW 1 en 2 te verstaan en te kan toepas.
- ‘n Kritiese beoordeling en vergelyking van stelselontwikkelingsmetodologieë te kan doen.

Vaardighede:

Studente sal stelselontwikkelingsmetodologieë krities kan beoordeel, en ‘n geskikte metodologie vir ‘n bepaalde projek kan aanbeveel. Studente sal stelselontwikkelings-metodologieë kan toepas en ‘n groot projek daarmee kan ontwikkel.

MODULE OUTCOMES

Goal:

To present the learner with a good knowledge and practical experience of the different system development methodologies.

Knowledge:

At the end of the semester the students will have a sound knowledge of the different system development methodologies. This knowledge includes: System development methods, underlying approaches on which the system development methodologies are based, development process models which are used in system development methods, development techniques and tools which are used in system development methods.

After completing this module students ought to be able to:

- Define and explain what Information System Engineering is
- Define and explain system development methodologies.
- Explain the acceptance of system development methodologies in practice.
- Understand and apply STRADIS (Structured Analysis, Design, and Implementation of Information Systems).
- Understand and apply IE (Information Engineering).
- Understand and apply RUP (Rational Unified Process).
- Understand and apply XP (Extreme Programming).
- Understand and apply SSM (Soft Systems Methodology).
- Understand and apply ETHICS (Effective Technical and Human Implementation of Computer-based Systems.
- Understand and apply MULTIVIEW 1 and 2.
- Give a critical review and comparison of the system development methodologies.

Skills:

Students will be able to give a critical review of system development methodologies to recommend a suitable methodology for a given project. Students will be able to apply and to implement with these skills system development methodologies for a large project.

STUDIE-IKONE

Antwoorde/oplossings

Individuele oefening

Bestudeer nou die volgende gedeelte/verduideliking/bespreking aandagtig

Geskatte studietyd

Bestudeer die aangeduide materiaal in die handboek/artikel, ens

Uitkomste

AKSIEWERKWOORDE

Vrae, hetsy in toetse of eksamens, bevat altyd sekere sleutel- of aksiewerkwoorde. Studente moet weet wat die aksiewoorde beteken en wat van hulle by die beantwoording van die vraag verlang word. Met die oog hierop word hier onder ’n kort lysie van sulke woorde verskaf. 'n Kort omskrywing van aksiewerkwoorde met 'n hoë gebruiksfrekwensie in vraestelle sal die lysie nog meer toepaslik maak.

- **Noem**

Hier word net die feite kort en saaklik neergeskryf.

- **Beskryf**

Hier word prestasie op die kennisvlak verwag. Eienskappe, feite of resultate word op 'n logiese, goed gestruktureerde wyse weergegee. Geen kommentaar of beredenering is nodig nie.

- **Definieer**

Kennisreproduksie word vereis. Dit is 'n duidelike, kernagtige en gesaghebbende beskrywing van 'n begrip sodat die betekenis daarvan duidelik blyk.

- **Gee 'n oorsig**

'n Groot volume kennis moet op 'n logiese en sistematiese wyse saamgevat en weergegee word, sonder dat die essensie van die saak verlore raak.

- **Verduidelik**

Die saak word eenvoudig gestel sodat die leser dit sal verstaan. Hier moet van illustrasies, beskrywings en voorbeelde gebruik gemaak word, terwyl ook redes vir uitsprake en resultate gegee word.

- **Bewys**

Staaf die feite deur die logiese aanvoering van aanvaarbare feite.

STUDY ICONS


Answers/Solutions


Individual exercise


Now study the following part/explanation/discussion attentively


Estimated study time


Study the indicated material in the handbook/article etc.


Outcomes

ACTION VERBS

Questions, either in tests or exams, always contain certain keywords or action verbs.

Students should know the meaning of action words and what is expected from them in answering the questions. Keeping this in mind a short list follows hereunder. A short description of action verbs with a high frequency in questionnaires will increase the appropriateness.

- **Name**

Only the facts are stated here short and sweetly.

- **Describe**

Use of the knowledge base is expected. Characteristics, facts or results should be stated in a logical, well structured manner. No comments or reasoned exposition is necessary.

- **Define**

Knowledge reproduction is a requisite. It is a clear, pithy and authoritative description of a concept in order that the meaning becomes clear.

- **Give a summary**

A large volume of knowledge must be collated and reproduced in a logical and systematic manner without losing the essence of the matter.

- **Explain**

The matter is simply put so that the reader will understand it. Illustrations, descriptions and examples should be used and reasons for the findings and results given.

- **Proof**

Confirm the facts through logical reasoning of acceptable facts.

- **Vergelyk**

Hierdie vraag moet versigtig hanteer word. Die een saak moenie eers beskryf of bespreek word en dan later die volgende nie. Feite, gebeure of probleme word teenoor mekaar gestel en ooreenkomste en verskille word na vore gebring.

- **Bespreek**

Hierdie tipe vraag veronderstel insig en onderskeidingsvermoë wanneer 'n toepassing en beoordeling gemaak word. Hier word die verskeie aspekte van die saak of stelling op analitiese wyse ondersoek en gestel.

- **Ontleed/analiseer**

Die inhoud word in dele of elemente verdeel en bespreek. Oorsake en gevolge word bepaal en onderlinge verwantskappe word vasgestel.

- **Evalueer**

'n Saak behoort aan die hand van sekere kriteria beoordeel te word. 'n Waarde-oordeel word oor die saak uitgespreek en dit moet gemotiveer word.

Dosente sal waar nodig in die klasse ook verdere toeligting oor hierdie (en ander tersaaklike terme) verskaf. Vir u eie doel is dit egter belangrik om vooraf te weet wat van u verwag word as so 'n aksiewerkwoord in 'n vraestel voorkom.

WAARSKUWING TEEN PLAGIAAT

WERKSTUKKE IS INDIVIDUELE TAKE EN NIE GROEPAKTIWITEITE NIE (TENSY DIT UITDRUKLIK AANGEDUI WORD AS 'N GROEPAKTIWITEIT)

Kopiëring van teks van ander leerders of uit ander bronne (byvoorbeeld die studiegids, voorgeskrewe studiemateriaal of direk vanaf die internet) is **ontoelaatbaar** – net kort aanhalings is toelaatbaar en slegs indien dit as sodanig aangedui word.

U moet bestaande teks **herformuleer** en u **eie woorde** gebruik om te verduidelik wat u gelees het. Dit is nie aanvaarbaar om bestaande teks/stof/inligting bloot oor te tik en die bron in 'n voetnoot te erken nie – u behoort in staat te wees om die idee of begrip/konsep weer te gee sonder om die oorspronklike skrywer woordeliks te herhaal.

Die doel van die opdragte is nie die blote weergee van bestaande materiaal/stof nie, maar om vas te stel of u oor die vermoë beskik om bestaande tekste te integreer, om u eie interpretasie en/of kritiese beoordeling te formuleer en om 'n kreatiewe oplossing vir bestaande probleme te bied.

Wees gewaarsku: Studente wat gekopieerde teks indien sal 'n nulpunt vir die opdrag ontvang en dissiplinêre stappe mag deur die Fakulteit en/of die Universiteit teen sodanige studente geneem word. Dit is ook onaanvaarbaar om iemand anders se werk vir hulle te doen of iemand anders in staat te stel om u werk te kopieer – moet dus nie u werk uitleen of beskikbaar stel aan ander nie!

- **Compare**

This type of question must be handled with due care. Do your comparison throughout simultaneously while describing or discussing relevant matters. Facts, occurrences or problems are situated against each other and then similarities and differences put to the fore.

- **Discuss**

This type of question presumes insight and power of discrimination whenever an implementation or judgement is made. Through the use of analytical ways the different aspects of the case or views are investigated and presented.

- **Dissect/analyse**

The contents are divided in parts or elements and discussed. Causes and effects are determined and mutual relationships are fixed.

- **Evaluate**

It is expected that through evaluation the case be judged by certain criteria. A value-judgment is made of the case and it must be substantiated.

In the classroom environment lecturers will, where necessary, give further explanations on this and other relevant terms. For your own benefit it is important to know what is expected from you when an action verb appears in an exam paper.

WARNING AGAINST PLAGIARISM

ASSIGNMENTS ARE INDIVIDUAL TASKS AND NOT GROUP ACTIVITIES. (UNLESS EXPLICITLY INDICATED AS GROUP ACTIVITIES)

Copying of text from other learners or from other sources (for instance the study guide, prescribed material or directly from the internet) is **not allowed** – only brief quotations are allowed and then only if indicated as such.

You should **reformulate** existing text and use your **own words** to explain what you have read. It is not acceptable to retype existing text and just acknowledge the source in a footnote – you should be able to relate the idea or concept, without repeating the original author to the letter.

The aim of the assignments is not the reproduction of existing material, but to ascertain whether you have the ability to integrate existing texts, add your own interpretation and/or critique of the texts and offer a creative solution to existing problems.

Be warned: students who submit copied text will obtain a mark of zero for the assignment and disciplinary steps may be taken by the Faculty and/or University. It is also unacceptable to do somebody else's work, to lend your work to them or to make your work available to them to copy – be careful and do not make your work available to anyone!

Leereenheid 1

INLEIDING TOT INLIGTINGSTELSEL-ONTWIKKELING

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 1 en 2 van die handboek gebaseer, asook op die artikel wat in Bylae A voorkom.

Aan die einde van hierdie leereenheid behoort u

- die omgewing en konteks waarbinne inligtingstelselontwikkeling plaasvind, te verstaan,
- faktore wat die omgewing en konteks beïnvloed waarbinne inligtingstelselontwikkeling plaasvind, te kan beskryf,
- die menslike aspekte wat inligtingstelselontwikkeling beïnvloed, te kan beskryf,
- die organisatoriese aspekte wat inligtingstelselontwikkeling beïnvloed, te kan beskryf,
- die professionele aspekte wat inligtingstelselontwikkeling beïnvloed, te kan beskryf,
- ‘n inligtingstelsel te kan definieer,
- te kan verduidelik waarom daar ‘n behoefte aan ‘n metodologie is,
- ‘n stelselontwikkelingsmetodologie te kan definieer, en
- die komponente van ‘n stelselontwikkelingsmetodologie te kan verduidelik.

Study Unit 1

INTRODUCTION INTO INFORMATION SYSTEM DEVELOPMENT

You ought to be able to work successfully through this study unit in about three hours.

This study unit is based on chapters 1 and 2 of the handbook as well as on the article that appears as annexure A.

You should, at the end of this study unit, be able:

- To understand the environment and context wherein information system development takes place;
- to describe the factors wherein information system development takes place;
- to describe the human influence on information system development;
- to describe the organisational aspects of information system development;
- to describe the professional aspects of information system development;
- to define an information system;
- to explain why there is a need for a methodology;
- to define a system development methodology; and
- to be able to explain the components of a system development methodology.

Berei voor in die handboek:

- Bespreking van wat ‘n inligtingstelsel is en voorbeelde daarvan op bl 3-7. Maak seker van die volgende terme:
    - E Inligtingstelsel
    - E Formele inligtingstelsel
    - E Rekenaargebaseerde inligtingstelsel
- Bespreking van die omgewing en konteks waarbinne inligtingstelselontwikkeling plaasvind op bl 7-11. Hierdie bespreking sluit faktore in wat die omgewing en konteks beïnvloed:
    - E Globale ekonomie
    - E Digitale ekonomie
    - E Elektroniese handel
    - E Nie-kommersiële impakte
    - E Verandering
- Die *menslike* aspekte wat inligtingstelselontwikkeling beïnvloed op bl 11-14. Maak seker u kan die verskillende rolspelers identifiseer.
- Die *organisatoriese* aspekte wat inligtingstelselontwikkeling beïnvloed op bl 14.
- Die *professionele* aspekte wat inligtingstelselontwikkeling beïnvloed op bl.14-19.
- Sleutelkonsepte in inligtingstelselontwikkeling op bl 21-23. Maak seker u kan verduidelik wat elkeen van die volgende terme behels:
    - E Data
    - E Inligting
    - E Kennis
    - E Stelsel
    - E Substelsel
    - E Inligtingstelsel
- Behoefte na ‘n metodologie op bl 23-24.
- Inligtingstelselontwikkeling-metodologie op bl 24-26. Bestudeer ook die artikel in Bylae A. Maak seker u kan die volgende doen:
    - E Verduidelik wat ‘n inligtingstelselontwikkeling-metodologie (ISOM – in Engels SOM) is.
    - E Onderskei tussen die verskillende komponente van ‘n ISOM, nl
        - die filosofiese benadering waarop dit gebaseer is,
        - die prosesmodel wat gevolg word,

Prepare from the handbook:

- Discuss what an information system is and furnish examples (pp 3-7). Be sure on the following terms:
  - E Information system;
  - E Formal information system; and
  - E Computer based information system
- Discussion of the environment and context wherein system development systems take place on pp 7-11. This discussion includes factors which influence the environment and context:
  - E Global economy;
  - E Digital economy;
  - E Electronic business;
  - E Non-commercial impacts; and
  - E Change
- The *humanistic* aspect that influences information system development is on pp 11 - 14. Ensure that you can identify the different role players.
- The *organisational* aspects that influence information system development on p 14.
- The p*rofessional* aspects that influence information system development on pp 14-19.
- Key concepts in respect of information system development on pp 21-23. Ensure that you can explain what each of the following terms comprise:
  - E Data;
  - E Information;
  - E Knowledge;
  - E System;
  - E Sub-system; and
  - E Information system.
- The need for a methodology pp 23-24
- Information system development methodology on pp 24-26. Also study the article in annexure A. Ensure that you can do the following:
  - E Explain what an information System Development Methodology (SDM) is.
  - E Differentiate between the different components of a SDM, i.e.
    - The philosophical approach on which it is based;
    - the process model that should be followed;
    - the method itself; and

- die metode self,
- die tegnieke en hulpmiddels wat dit gebruik.

E Verduidelik wat die doelwitte van 'n ISOM is.

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 4 op bl 20. (10)

b) Vraag 2 op bl 27. (30)

Om hierdie vraag te kan beantwoord, sal u addisionele bronne in die biblioteek moet raadpleeg. Onthou om reg te verwys en die bronnelys in te sluit.

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

- the techniques and resources that it uses.

E Explain the aims of a SDM.

Do the following exercise after studying the chapter and hand it in at the beginning of the next class. Do the following exercise after studying the chapter and submit at the beginning of the following class opportunity:

a) Question 4 on p 20. (10)

b) Question 2 on p 27. (30)

To answer this question you will need to access and consult additional sources in the library. Remember to correctly refer to sources and to include the source list.

The memorandum will be discussed as soon as the lecturer hands out the assignments.

Leereenheid 2

LEWENSIKLUSBENADERING

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 3 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u

- die verskillende fases in die inligtingstelselontwikkelings-lewensiklus te kan beskryf en dit te kan toepas op IT-projekte,
- te kan beoordeel of die inligtingstelselontwikkelings-lewensiklus ‘n ISOM is,
- die potensiële sterk punte van die inligtingstelselontwikkelings-lewensiklus te kan bespreek,
- die potensiële swak punte van die inligtingstelselontwikkelings-lewensiklus te kan bespreek.

Berei voor in die handboek:

- Die verskillende fases in die inligtingstelselontwikkelings-lewenssiklus op bl 31-35. Dit sluit in:

Study Unit 2

LIFE CYCLE APPROACH

You ought to work through this study unit successfully in approximately three hours.

This study unit is based on Chapter 3 of the handbook.

You should, at the end of this study unit, be able to:

- Describe the different phases in the information system developments life cycle and to apply it to IT projects;
- judge whether the information system developments life cycle is a SDM;
- discuss the potential strengths of the information system developments life cycle; and
- discuss the potential weaknesses of the information system developments life cycle.

  E Uitvoerbaarheidstudie.
  E Stelselondersoek.
  E Stelselanalise.
  E Stelselontwerp.
  E Implementering.
  E Hersiening en onderhoud.

- Bespreking van die inligtingstelselontwikkelings-lewenssiklus met betrekking tot die komponente van ‘n ISOM op bl 35-38.
- Die potensiële sterk punte van die inligtingstelselontwikkelings-lewenssiklus op bl 38.
- Die potensiële swak punte van die inligtingstelselontwikkelings-lewenssiklus op bl 38-43. Dit sluit in:

  E Faal om aan bestuur se behoeftes te voldoen.
  E Modelle van prosesse is onstabiel.
  E Uitvoer-gedrewe ontwerp lei tot onbuigbaarheid.
  E Gebruikersontevredenheid.
  E Probleme met dokumentasie.
  E Gebrek aan kontrole.
  E Onvolledige stelsels.
  E Toepassingsagterstand.
  E Onderhoudwerklading.
  E Probleme met die “ideale” benadering.
  E Klem op “harde” aspekte.
  E Aanname van “groen veld”-ontwikkeling.

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 3 op bl 44. (20)

Om hierdie vraag te kan beantwoord, sal u addisionele bronne in die biblioteek moet raadpleeg. Onthou om reg te verwys en die bronnelys in te sluit.

b) U ontvang hierdie week u praktiese projek. Raak solank vertroud met die inhoud daarvan.

Prepare by use of the handbook:

- The different phases in the information system development life cycle on pp 31-35 This includes:
  - E Feasibility study;
  - E System investigation;
  - E System analysis;
  - E System design;
  - E Implementation; and
  - E Reviewing and maintenance.
- Discussion of the information system development life cycle in terms of the components of a SDM on pp 35-38.
- The potential strengths of the information system development life cycle on p 38.
- The potential weaknesses of the information system development life cycle on pp 38-43. This includes:
  - E Failure to meet managements needs;
  - E The models of the processes are unstable;
  - E Output driven designs lead to inflexibility;
  - E User dissatisfaction;
  - E Problems with documentation;
  - E Lack of control;
  - E Incomplete systems;
  - E Application backlog;
  - E Maintenance work load;
  - E Problems with the "ideal" approach;
  - E Emphasis on hard thinking; and
  - E Assumption of "green field" development.

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

Do the following exercise after studying the chapter and submit it at the following class opportunity:

a) Question 3 on p 44. (20)

In order to answer this question you will have to consult additional sources in the library. Remember to refer correctly and to include the source list.

b) This week you will receive your practical project. Familiarise yourself with the contents.

The memorandum is discussed when the lecturer hands out the assignments.

Leereenheid 3

PROSESGEORIËNTEERDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 6, 12 en 20 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u

- die STRADIS-metodologie te kan bespreek ten opsigte van die komponente van ‘n ISOM,
- die STRADIS-metodologie te kan toepas en ‘n projek daarmee te kan ontwikkel.

U vind die bespreking van hierdie metodologie in verskeie hoofstukke in die handboek. Gebruik die onderstaande tabel tydens u voorbereiding:

Study Unit 3

PROCESS ORIENTATED SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work successfully through this study unit in about three hours.

This study unit is based on chapters 6, 12, and 20 of the handbook.

You should, at the end of this study unit, be able to

- discuss the STRADIS methodology in terms of the components of a SDM
- apply the STRADIS methodology and to develop a project by the use thereof.

You will find the discussion of the methodology in various chapters of the handbook. Use the under mentioned list during your preparation:

STRADIS-metodologie		
Stelselontwikkelingsmetode	Structured Analysis, Design and Implementation of Information Systems (STRADIS)	Hoofstuk 20 pp 395 - 402
Filosofiese benadering	Prosesmodellering: Funksionele dekomposisie	Hoofstuk 6 pp 107-111
Prosesmodel	Fases (fokus hoofsaaklik op analise, minder op ontwerp, en byna glad nie op implementering nie). Kombinasie van tegnieke.	
Tegnieke	Datavloeidiagramme Datastruktuurdiagramme Besluitnemingsbome Besluitnemingstabelle Gestruktueerde Engels Normalisasie	Hoofstuk 12 pp 242-272

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 1, 2, en 3 op bl 272. (30)

b) Vraag 3 op bl 418.

Gebruik STRADIS as die metodologieë wat bespreek word. (10)

c) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

	STRADIS methodology	
System development method	Structured Analysis, Design and Implementation of Information Systems (STRADIS)	Chapter 20 pp 395 - 402
Philosophical approach	Process modelling: Functional decomposition	Chapter 6 pp 107-111
Process model	Phases (focus mainly on analysis, less on design, and hardly at all on implementation). Combination of techniques.	
Techniques	Data flow diagrammes Data Structure Diagrammes Decision trees Decision tables Structured English Normalisation	Chapter 12 pp 242-272

Do the following exercises after studying the chapter and submit it at the beginning of the following class opportunity:

a) Question 1, 2, and 3 on p 272. (30)

b) Question 3 on p 418.
Use STRADIS as the methodologies discussed. (10)

c) Practical project

The memorandum is discussed when the lecturer hands out the assignments again.

Leereenheid 4

DATA- (GEMENGDE) GEORIËNTEERDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 4, 6, 11, 12, 15 en 21 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u:

- Die IE (Information engineering) -metodologie te kan bespreek ten opsigte van die komponente van 'n ISOM,
- Die IE-metodologie te kan toepas en 'n projek daarmee te kan ontwikkel.

U vind die bespreking van hierdie metodologie in verskeie hoofstukke in die handboek. Gebruik die onderstaande tabel tydens u voorbereiding:

Study Unit 4

DATA (BLENDED) ORIENTATED SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work successfully through this study unit in about three hours.

This study unit is based on chapters 4, 6, 11, 12, 15 and 21 of the handbook

You should, at the end of this study unit, be able to:

- Discuss the IE methodology in terms of the components of a SDM; and
- apply the IE methodology and to be able to develop a project by the use thereof.

The discussion of this methodology will be found in various chapters in the handbook. Use the following table when doing your preparation:

IE-metodologie		
Stelselontwikke-lingsmetode	Information Engineering (IE)	Hoofstuk 21 pp434-446
Filosofiese benadering	Strategiese Inligtingstelsels Datamodellering	Hoofstuk 4 pp 53-62 Hoofstuk 6 pp111-113
Prosesmodel	Parallelle ontwikkeling	
Tegnieke	Entiteitmodellering Normalisasie Entiteitlewensilkus Besluitnemingsbome Besluitnemingstabelle Gestruktueerde Engels Aksiediagramme Kritiese suksesfaktore	Hoofstuk 11 pp 217-242 Hoofstuk 12 pp 243-272 Hoofstuk 15 pp 295-306

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 2 op bl 78. (5)

b) Vraag 2 op bl 117. (10)

c) Stel ‘n tabel op wat die verskille en ooreenkomste van STRADIS en IE aantoon. (20)

d) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

IE Methodology		
System development method	Information Engineering (IE)	Chapter 21 Pp 434-446
Philosophical approach	Strategic information systems Data modelling	Chapter 4 pp 53-62 Chapter 6 pp111-113
Process model	Parallel Development	
Techniques	Entity modelling Normalisation Entity life cycle Decision trees Decision tables Structured English Action diagrammes Critical success factors	Chapter 11 pp 217-242 Chapter 12 pp 243-272 Chapter 15 pp 295-306

Do the following exercise after studying the chapter and submit it at the following class opportunity:		
a)	Question 2 on p 78.	(5)
b)	Question 2 on p 117.	(10)
c)	Draw up a table showing the differences and similarities between STRADIS and IE	(20)
d)	Practical project	

The memorandum is discussed when the lecturer again hands out the assignments.

Leereenheid 5

OBJEKGEORIËNTEERDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 6, 8, 13,14 en 22 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u:

- Die RUP-metodologie te kan bespreek ten opsigte van die komponente van ‘n ISOM,
- Die RUP-metodologie te kan toepas en ‘n projek daarmee te kan ontwikkel.

U vind die bespreking van hierdie metodologie in verskeie hoofstukke van die handboek. Gebruik die onderstaande tabel tydens u voorbereiding:

Study Unit 5

OBJECT ORIENTATED SYSTEM DEVELOPMENT-METHODOLOGIES

You ought to be able to work successfully through this study unit in about three hours.

This study unit is based on Chapters 6, 8, 13, 14 and 22 of the handbook.

You should, at the end of this study unit, be able to:

- Discuss the RUP methodology in terms of the components of a SDM; and
- apply the RUP methodology as well as to develop a project using this method.

Several chapters of your handbook deal with this methodology. Use the following table when doing your preparation:

RUP-metodologie		
Stelselontwikke-lingsmetode	Rational Unified Process (RUP)	Hoofstuk 22 pp 460-467
Filosofiese benadering	Objekmodellering	Hoofstuk 6 pp 113-116 Hoofstuk 8 pp 161-163
Prosesmodel	Iteratief en inkrementeel Spiraal	
Tegnieke	Objekgeorienteerd UML (Unified modelling language): Klasdiagramme “Use case”-diagramme Interaksie-diagramme “Sequence”-diagramme “Statechart”-diagramme Aktiwiteitsdiagramme Pert-diagramme Gantt-diagramme	Hoofstuk 13 pp 273-288 Hoofstuk 14 pp 289-294

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 1 en 3 op bl 288. 20)

b) Vraag 2 en 3 op bl 468. (20)

c) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

RUP Methodology		
System development method	Rational Unified Process (RUP)	Chapter 22 pp 460-467
Philosophical approach	Object modelling	Chapter 6 pp113-116 Chapter 8 pp 161-163
Process model	Iterative and incremental Spiral	
Techniques	Object orientated UML (Unified modelling language): Class diagrammes Use case diagrams Interaction diagrams Sequence diagrams State chart diagrams Activity diagrams Pert charts Gantt charts	Chapter 13 pp 273-288 Chapter 14 pp 289-294

Do the following exercise after studying the chapter and submit it at the following class opportunity:

a) Question 1 en 3 on p 288. 20)

b) Question 2 and 3 on p 468. (20)

c) Practical project

The memorandum is discussed when the lecturer again hands out the assignments.

Leereenheid 6

RAD-GEORIËNTEERDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 7, 8 en 23 van die handboek gebaseer, asook op bylae B.

Aan die einde van hierdie leereenheid behoort u:

- Die XP-metodologie te kan bespreek ten opsigte van die komponente van ‘n ISOM,
- Die XP-metodologie te kan toepas en ‘n projek daarmee te kan ontwikkel.

U vind die bespreking van hierdie metodologie in verskeie hoofstukke van die handboek. Gebruik die onderstaande tabel tydens u voorbereiding:

Study Unit 6

RAD ORIENTED SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work successfully through this study unit in about three hours.

This study unit is based on Chapter 7, 8 and 23 of the handbook as well as Annexure B.

You should, at the end of this study unit, be able to:

- Discuss the XP methodology in terms of the components of a SDM; and
- apply the XP methodology as well as to develop a project using this method.

Several chapters of your handbook deal with this methodology. Use the following table when doing your preparation:

XP-metodologie		
Stelselontwikkelingsmetode	"Extreme programming" (XP)	Hoofstuk 23 pp 479-481 Bylae B
Filosofiese benadering	"Rapid Application Development" "Agile"-benadering (Objekgeorienteerd)	Hoofstuk 7 pp 119-150 Hoofstuk 8 pp 161-163
Prosesmodel	Inkrementele ontwikkeling Iteratiewe ontwikkeling Fases	
Tegnieke	Funksionele ontbinding "Timeboxing" Pareto-beginsel MoSCoW-reëls JAD-werksessies Prototipering ("Architectural Spike") "User stories" "Paired programming"	Hoofstuk 7 pp 128-134

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 2 op bl 149. (20)

Om hierdie vraag te kan beantwoord sal u addisionele bronne in die biblioteek moet raadpleeg. Onthou om reg te verwys en die bronnelys in te sluit.

b) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

XP Methodology		
System development method	Extreme programming (XP)	Chapter 23 pp 479-481 Annexure B
Philosophical approach	Rapid Application Development Agile approach (Object orientated)	Chapter 7 pp 119-150 Chapter 8 pp 161-163
Process model	Incremental development Iterative development Phases	
Techniques	Functional decomposition Time boxing Pareto principle MoSCoW rules JAD work sessions Prototyping (Architectural Spike) User stories Paired programming	Chapter 7 pp 128-134

Do the following exercise after studying the chapter and submit it at the following class opportunity:

a) Question 2 on p 149. (20)

In order to answer this question you will have to consult additional sources in the library. Remember to refer correctly and to include the source list.

b) Practical project.

The memorandum is discussed when the lecturer again hands out the assignments.

Leereenheid 7

MENSGEORIËNTEERDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 5, 16 en 24 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u:

- Die ETHICS-metodologie te kan bespreek ten opsigte van die komponente van 'n ISOM,
- Die ETHICS-metodologie te kan toepas en 'n projek daarmee te kan ontwikkel.

U vind die bespreking van hierdie metodologie in verskeie hoofstukke in die handboek. Gebruik die onderstaande tabel tydens u voorbereiding:

Study Unit 7

PEOPLE ORIENTED SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work successfully through this study unit in about three hours.

This study unit is based on Chapter 5, 16 and 24 of the handbook.

You should, at the end of this study unit, be able to:

- Discuss the ETHICS methodology in terms of die components of a SDM; and
- apply the ETHICS methodology and be able to develop a project.

Several chapters of your handbook deal with this methodology. Use the following table in your preparation:

ETHICS-metodologie		
Stelselontwikkelingsmetode	Effective technical and human implementation of computer-based systems (ETHICS)	Hoofstuk 24 pp487-496
Filosofiese benadering	Mens georiënteerd Sosiaal-tegnies	Hoofstuk 5 pp 79-108
Proses model	Stappe wat in siklusse uitgevoer word.	
Tegnieke	Rolspeler-analise JAD (Joint application development)	Hoofstuk 16 pp 307-314

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 1 op bl 108. (10)

b) Vraag 4 op bl 313. (10)

c) Vraag 1 en 3 op bl 505. (20)

Om hierdie vraag te kan beantwoord sal u addisionele bronne in die biblioteek moet raadpleeg. Onthou om reg te verwys en die bronnelys in te sluit.

d) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

ETHICS Methodology		
System Development Method	Effective technical and human implementation of computer-based systems (ETHICS)	Chapter 24 pp487-496
Philosophical approach	Human orientated Social technical	Chapter 5 pp 79-108
Process Model Proses model	Steps that are completed in cycles	
Techniques	Role player analysis JAD (Joint application development)	Chapter 16 pp 307-314

Do the following exercise after studying the chapter and hand in at the following class opportunity:

a) Question 1 on p 108. (10)

b) Question 4 on p 313. (10)

c) Question 1 en 3 on p 505. (20)

To be able to answer this question you will have to consult additional sources in the library. Remember to correctly show the additional sources and to include the source list.

d) Practical project

The memorandum is discussed when the lecturer again hands out the assignments.

Leereenheid 8

ORGANISASIE GEORIËNTEERDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 4, 10 en 25 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u:

- die SSM-metodologie te kan bespreek ten opsigte van die komponente van 'n ISOM,
- die SSM-metodologie te kan toepas en 'n projek daarmee te kan ontwikkel.

U vind die bespreking van hierdie metodologie in verskeie hoofstukke in die handboek. Gebruik die onderstaande tabel tydens u voorbereiding:

Study Unit 8

ORGANISATIONAL ORIENTATED SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work through this study unit successfully in about three hours.

This Study unit is based on Chapters 4, 10 and 25 of the handbook.

You should, at the end of this study unit, be able to:

- Discuss the SSM methodology in terms of the components of a SDM; and
- be able to apply the SSM methodology and to develop a project through the use of the aforementioned methodology

You will find the discussion of this methodology in various chapters in the handbook. Use the following table when doing your preparation:

SSM-metodologie		
Stelselontwikke-lingsmetode	Soft Systems Methodology (SSM)	Hoofstuk 25 pp 507-516
Filosofiese benadering	Stelselteorie	Hoofstuk 4 pp 51-78
Prosesmodel	Iteratiewe ontwikkeling in fases	
Tegnieke	“Rich pictures” “Root definitions” Konseptuele modelle	Hoofstuk 10

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid: a) Vraag 2 op bl 78. (10) b) Vraag 2 op bl 216. (20) c) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

	SSM Methodology	
System Development Method	Soft Systems Methodology (SSM)	Chapter 25 pp 507-516
Philosophical approach	Systems theory	Chapter 4 pp 51-78
Process model	Iterative development in phases	
Techniques	Rich pictures Root definitions Conceptual models	Chapter 10

Do the following exercise after studying the chapter and hand in at the following class opportunity:

a) Question 2 on p 78. (10)

b) Question 2 on p 216. (20)

c) Practical project

The memorandum will be discussed when the lecturer again hands out the assignments.

Leereenheid 9

RAAMWERKE

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 26 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u:

- Die verskil tussen stelselontwikkelings*raamwerke* en stelselontwikkelings*metodologieë* te verstaan,
- Die MULTIVIEW (1 en 2)-raamwerke te kan bespreek,
- Die MULTIVIEW (1 en 2)-raamwerke te kan toepas en ‘n projek daarmee te kan ontwikkel.
- Te kan verduidelik waarvoor die CMM-raamwerk gebruik word,
- Die CMM-raamwerk te kan bespreek.

Study Unit 9

FRAMEWORKS

You ought to be able to work through this study unit successfully in about three hours.

This study unit is based on Chapter 26 of the handbook.

You should, at the end of this study unit, be able to:

- Understand the difference between system development frameworks and system development methodologies;
- discuss the MULTIVIEW (1 and 2) frameworks;
- apply the die MULTIVIEW (1 en 2) frameworks and to be able to develop a project;
- explain what the CMM framework is used for; and
- discuss the CMM framework.

Berei voor uit die handboek:

- Die bespreking van MULTIVIEW 1 op bl 537-545. Let op die volgende:
  - E Fig 26.1 is ‘n goeie opsomming van die MULTIVIEW-raamwerk.
  - E Onderskei tussen die vyf fases:
    - Ontleding van menslike aktiwiteit.
    - Ontleding van inligting.
    - Ontleding en ontwerp van die sosio-tegniese aspekte.
    - Ontleding van die mens-rekenaar-koppelvlak.
    - Ontwerp van die tegniese aspekte.
- Die bespreking van MULTIVIEW 2 op bl 545-549. Let op die volgende:
  - E Maak seker dat u kan verduidelik hoe die twee weergawes van MULTIVIEW verskil. Fig. 26.8 gee ‘n handige opsomming.
  - E Fig. 26.10 is baie belangrik.
- Die bespreking van die CMM-raamwerk op bl 551-558. Maak seker u kan die vyf volwassenheidsvlakke onderskei.

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a)	Vraag 1 op bl 564.	(5)
b)	Vraag 2 op bl 564.	(20)
c)	Vraag 3 op bl 564.	(10)
d)	Praktiese projek	

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

Prepare from the handbook:

- The discussion of MULTIVIEW on pp 537-545. Please note the following:
  - E Fig 26.1 is a good summary of the MULTIVIEW framework.
  - E Differentiate between the five phases:
    - Analysis of human activity.
    - Analysis of information.
    - Analysis and design of the socio-technical aspects.
    - Design of the human-computer-interface.
    - Design of the technical aspects.
- The discussion of MULTIVIEW 2 on pp 545-549. Please note the following:
  - E Ensure that you can explain how the two variations of MULTIVIEW differ. Fig 26.8 is a handy summary.
  - E Fig. 26.10 is very important.
- The CMM framework is discussed on pp 551-558. Ensure that you can differentiate between the five maturity levels.

Do the following exercise after studying the chapter and hand in at the following class opportunity:	
a) Question 1 on p 564.	(5)
b) Question 2 on p 564.	(20)
c) Question 3 on p 564.	(10)
d) Practical project	

The lecturer will discuss the memorandum when the assignments will be distributed again.

Leereenheid 10

INLIGTINGSTELSELONT-WIKKELING SONDER METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 9, 17, 18 en 19 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u

- verskeie vorme van eksterne ontwikkeling te kan bespreek,
- die voordele van die gebruik van tegnieke te kan verduidelik,
- te kan verduidelik hoe persone se begrip van ‘n probleem beïnvloed word deur die tegniek of tegnieke wat tot hulle beskikking is,
- tegnieke te kan klassifiseer gebaseer op hulle objektiewe en subjektiewe eienskappe,
- te kan verduidelik wat bedoel word met sagtewarehulpmiddels,
- te kan verduidelik wat bedoel word met sagtewarehulpmiddelversamelings (CASE tools),
- die voordele en probleme in die gebruik van sagtewarehulpmiddelversamelings (CASE tools) te kan verduidelik.

Study Unit 10

INFORMATION SYSTEM DEVELOPMENT WITHOUT ETHODOLOGIES

You ought to be able to work through this study unit successfully in about three hours.

This Study unit is based on Chapters 9, 17, 18 and 19 of the handbook

You should, at the end of this study unit, be able to:

- Discuss various forms of external development;
- explain the benefits of the usage of techniques;
- explain how peoples' understanding of a problem can be influenced through the technique(s) that is available to them;
- classify techniques based on their objective and subjective characteristics;
- explain what is meant by software tools;
- explain what is meant by CASE tools; and
- discuss the advantages and problems of the use of CASE tools.

Berei voor in die handboek:

- Die verskillende vorme van eksterne ontwikkeling op bl 175-192. Maak seker u kan die volgende begrippe goed verduidelik:
  - E Toepassingspakkette
  - E "Open Source"-sagteware
  - E ERP (Enterprise resource planning)-stelsels
  - E "Outsourcing" en "Offshoring"
- Voordele van tegnieke en eienskappe wat gebruik kan word om tegnieke te klassifiseer op bl 316-317.
- Hoe persone se begrip van 'n probleem beïnvloed word deur die tegniek of tegnieke wat tot hulle beskikking is op bl 317-321.
- Twee dimensionele klassifikasie van tegnieke op bl 321-324.
- Sagtewarehulpmiddels op bl 331-358.
- Sagtewarehulpmiddelversamelings (CASE tools) op bl 361-375.
- Voordele en probleme tydens die gebruik van sagtewarehulpmiddelversamelings (CASE tools) op bl 375-386.

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 2 en 3 op bl 193. (10)

b) Vraag 3 op bl 326. Gebruik die praktiese projek waarmee u besig is om hierdie vraag te beantwoord. (20)

c) Vraag 5 op bl 387. Gebruik u ervaring met ORACLE Designer om hierdie vraag te beantwoord. Gee 'n waarde tussen en 1 en 5 vir die beoordeling van elke teoretiese voordeel. Die skaal is soos volg:

   1 = Stem glad nie saam nie.

   2 = Stem in 'n mindere mate saam.

   3 = Neutraal

   4 = Stem in 'n groot mate saam

   5 = Stem heeltemal saam

   Motiveer telkens u beoordeling. Bespreek ook die probleme wat u ervaar het met die gebruik van ORACLE Designer. (40)

d) Praktiese projek

Prepare from the handbook:

- The different forms of external development on pp 175 – 192. Ensure that you can fully explain the following concepts:
  - E Application packages;
  - E Open Source software;
  - E ERP (Enterprise Resource Planning) systems; and
  - E Outsourcing and Offshoring
- Advantages of techniques and characteristics that may be used to classify techniques on pp 316-317.
- How persons' conception of a problem can be influenced through a technique(s) that is available to them on pp 317-321.
- Two-dimensional classification of techniques on pp 321-324.
- Software tools on pp 331-358.
- CASE tools on pp 361-375.
- Advantages and problems during the use of CASE tools on pp 375-386.

Do the following exercise after studying the chapter and hand it in at the following class opportunity:

a) Question 2 and 3 on p 193. (10)

b) Question 3 on p 326. Use your practical project to answer this question. (20)

c) Question 5 on p 387. Use your experience with ORACLE Designer to answer this question. Award a mark between 1 and 5 for the judging of each theoretical advantage. Use this scale:

1 = Do not agree at all.

2 = Agree to a lesser extent.

3 = Neutral.

4 = Agree to a greater extent.

5 = Agree fully.

Motivate your judgements. Also discuss the problems you encountered with the use of ORACLE Designer. (40)

d) Practical project.

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

The memorandum is discussed when the lecturer again hands out the assignments.

Leereenheid 11

RELEVANTE KWESSIES RAKENDE STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 27 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u

- ‘n volledige definisie van ‘n stelselontwikkelingsmetodologie (ISOM) te kan verskaf,
- te kan verduidelik watter voordele die gebruik van ‘n ISOM kan inhou,
- te kan verduidelik wat gebeur tydens die aanneming van ‘n ISOM in die praktyk,
- die evolusie en ontwikkeling van ISOM te kan bespreek.

Study Unit 11

RELEVANT MATTERS CONCERNING SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work through this study unit successfully in about three hours.

This study unit is based on Chapter 27 of the handbook

You should, at the end of this study unit, be able to:

- Give a full definition of system development methodologies;
- explain what advantages the use of SDM may entail;
- explain what happens during the acceptance of SDM in practice; and
- discuss the evolution and development of SDM.

Berei voor in die handboek:

- Die definisie van ‘n ISOM op bl 567-570. Maak seker dat u duidelik kan onderskei tussen
  - E die verskillende komponente van ‘n ISOM
  - E die term “metode” en “metodologie”.
- Voordele tydens die gebruik van ‘n ISOM op bl 570-572. Dit sluit in:
  - E ‘n Beter eindproduk
  - E ‘n Beter ontwikkelingsproses
  - E ‘n Gestandaardiseerde proses.
- Die aanneming van ‘n ISOM in die praktyk op bl 572-576.
- Evolusie en ontwikkeling van ISOM op bl 576-589. Maak seker u kan duidelik onderskei tussen die volgende eras:
  - E Pre-metodologie-era
  - E Vroeëmetodologie-era
  - E Metodologie-era
  - E Era van metodologie-herassessering.

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Vraag 1 en 4 op bl 590. (20)

b) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

Prepare from the handbook:

- The definition of a SDM on pp 567-570. Ensure that you clearly differentiate between:
  - E The different components of a SDM; and
  - E the term "method" and "methodology".
- Advantages during the use of a SDM on pp 570-572. This includes:
  - E A better end product;
  - E a better development process; and
  - E a standardised process.
- The adoption of a SDM in practice on pp 572-576.
- The evolution and development of SDM on pp 576-589. Ensure that you can clearly differentiate between the following eras:
  - E Pre-methodology era;
  - E Early methodology era;
  - E Methodology era; and
  - E The era of methodology reassessment.

Do the following exercise after studying the chapter and hand it in at the following class opportunity:

a) Question 1 en 4 on p 590. (20)

b) Practical project

The memorandum will be discussed when the lecturer again hands out the assignments.

Leereenheid 12

VERGELYKING VAN STELSELONTWIKKELING-METODOLOGIEË

U behoort ongeveer drie uur te neem om hierdie leereenheid suksesvol deur te werk.

Hierdie leereenheid is op Hoofstuk 28 van die handboek gebaseer.

Aan die einde van hierdie leereenheid behoort u:

- Redes te kan verskaf waarom metodologieë vergelyk word,
- Kriteria te verskaf en te verduidelik waarvolgens ISOM geassesseer kan word,
- Die raamwerk van Avison en Fitzgerald, waarvolgens ISOM vergelyk kan word, te ken en te kan toepas.

Berei voor in die handboek:

- Die kwessies rakende die vergelyking van metodologieë bl 591-597. Maak seker dat u die “ideale-tipe”-kriteria kan bespreek waarvolgens metodologieë beoordeel kan word.

Study Unit 12

COMPARISON OF SYSTEM DEVELOPMENT METHODOLOGIES

You ought to be able to work through this study unit successfully in about three hours.

This study unit is based on Chapter 28 of the handbook

You should, at the end of this study unit, be able to:

- Give reasons why methodologies are compared;
- provide criteria and to explain in what manner SDM can be assessed; and
- to know and apply the framework of Avison and Fitzgerald whereby SDM can be compared.

- Raamwerk vir die vergelyking van metodologieë op bl 597-603. Hierdie raamwerk is baie belangrik en bestaan uit die volgende dele:
    - E Filosofie
    - E Model
    - E Tegnieke en hulpmiddels
    - E Omvang
    - E Uitvoere
    - E Praktyk
    - E Produk
- Voorbeeld waar die raamwerk toegepas word op bl 604-613.

Doen die volgende oefeninge nadat u die hoofstuk bestudeer het en dien dit in aan die begin van die volgende klasgeleentheid:

a) Pas die raamwerk toe wat in hierdie hoofstuk bespreek is en vergelyk die STRADIS-, IE-, RUP-, XP-, ETHICS- en SSM-metodologieë. (100)

b) Praktiese projek

Die memorandum word bespreek wanneer die dosent die opdragte weer uitdeel.

Prepare in the handbook:

- The matters concerning the comparison of methodologies pp 591-597. Ensure that you can discuss the ideal type criteria whereby methodologies can be judged.
- Framework for the comparison of methodologies on pp 597-603. This framework is very important and it contains the following sections:
  - E Philosophy;
  - E Model;
  - E Techniques and tools;
  - E Scope;
  - E Outputs;
  - E Practice; and
  - E Product
- An example where the framework is applied on pp 604-613.

Do the following exercise after studying the chapter and hand it in at the following class opportunity:

a) Apply the framework which is discussed in this chapter and compare the STRADIS, IE, RUP, XP, ETHICS and SSM methodologies.

b) Practical project.

The memorandum will be discussed when the lecturer again hands out the assignments.

BYLAE A

Factors that affect the use and acceptance of systems development methodologies by system developers

Dr Magda Huisman

North-West University – Potchefstroom Campus

School for Computer Science, Statistics and Maths

North-West University – Potchefstroom Campus

Potchefstroom, South Africa

Email: rkwhmh@puknet.puk.ac.za

Abstract

In this study fourteen factors affecting the use and acceptance of systems development methodologies by individual systems developers were investigated. The results show that relative advantage, compatibility and trialability of a systems development methodology, an individual's experience in systems development and his/her experience in systems development methodologies, management support and peer developer support, and uncertainty about the continued existence of the IS department significantly influence the deployment of systems development methodologies.

Keywords

Systems development, systems development methodologies, use, acceptance, developers.

INTRODUCTION

Systems development methodologies (SDM) have formed one of the most intensive research topics in Information Systems (IS) and Software Engineering. In 1994 Jayaratna (1994) estimated the number of methodologies to be of order 1000, and since then many more systems development methodologies have been developed. It is generally assumed that systems development methodologies are used in practice (Saeki, 1998), and there exists a widespread belief that adherence to systems development methodologies (SDM) is beneficial to an organization (Fitzgerald, 1996; Hardy et al., 1995). This belief is manifested in the pressure that practitioners face today to use SDM (Fitzgerald, 1996). Despite the high investment in the development of SDM and the pressure to use it, their practical usefulness is still a controversial issue (Fitzgerald, 1996; Introna and Whitley, 1997; Nandhakumar and Avison, 1999). Many organisations claim that they do not use any methodologies (Huisman and Iivari, 2003; Hardy et al., 1995; Chatzoglou and Macaulay, 1996; Fitzgerald, 1998). Other organizations report that they are adapting SDM (Fitzgerald et al., 2003; Huisman and Iivari, 2003; Hardy et al., 1995; Russo et al., 1996), while others are using it with positive results (Chatzoglou and Macauly, 1996; Rahim et al., 1998). Apart from this, we do not know why SDM are used and what factors influence its use and effectiveness.

A decision by IS management to adopt SDM in an IS department does not guarantee that all developers will use the methodology, or that they will use it to its full potential. The purpose of this paper is to determine which factors influence the individual use and acceptance of SDM. Researchers emphasise that a distinction must be made between the adoption and acquisition of technology at the organisational level and its adoption and implementation at the individual level (Fichman, 1992; McChesney and Glass, 1993; Senn and Wynekoop,

ANNEXURE A

Factors that affect the use and acceptance of systems development methodologies by system developers

Dr Magda Huisman

North-West University – Potchefstroom Campus

School for Computer Science, Statistics and Maths

North-West University – Potchefstroom Campus

Potchefstroom, South Africa

Email: rkwhmh@puknet.puk.ac.za

Abstract

In this study fourteen factors affecting the use and acceptance of systems development methodologies by individual systems developers were investigated. The results show that relative advantage, compatibility and trialability of a systems development methodology, an individual's experience in systems development and his/her experience in systems development methodologies, management support and peer developer support, and uncertainty about the continued existence of the IS department significantly influence the deployment of systems development methodologies.

Keywords

Systems development, systems development methodologies, use, acceptance, developers.

INTRODUCTION

Systems development methodologies (SDM) have formed one of the most intensive research topics in Information Systems (IS) and Software Engineering. In 1994 Jayaratna (1994) estimated the number of methodologies to be of order 1000, and since then many more systems development methodologies have been developed. It is generally assumed that systems development methodologies are used in practice (Saeki, 1998), and there exists a widespread belief that adherence to systems development methodologies (SDM) is beneficial to an organization (Fitzgerald, 1996; Hardy et al., 1995). This belief is manifested in the pressure that practitioners face today to use SDM (Fitzgerald, 1996). Despite the high investment in the development of SDM and the pressure to use it, their practical usefulness is still a controversial issue (Fitzgerald, 1996; Introna and Whitley, 1997; Nandhakumar and Avison, 1999). Many organisations claim that they do not use any methodologies (Huisman and Iivari, 2003; Hardy et al., 1995; Chatzoglou and Macaulay, 1996; Fitzgerald, 1998). Other organizations report that they are adapting SDM (Fitzgerald et al., 2003; Huisman and Iivari, 2003; Hardy et al., 1995; Russo et al., 1996), while others are using it with positive results (Chatzoglou and Macauly, 1996; Rahim et al., 1998). Apart from this, we do not know why SDM are used and what factors influence its use and effectiveness.

A decision by IS management to adopt SDM in an IS department does not guarantee that all developers will use the methodology, or that they will use it to its full potential. The purpose of this paper is to determine which factors influence the individual use and acceptance of SDM. Researchers emphasise that a distinction must be made between the adoption and acquisition of technology at the organisational level and its adoption and implementation at

1995; Dietrich *et al.,* 1997; Lai and Guynes, 1997). In terms of Rogers (1995), systems development methodologies are contingent innovations with organisations as primary adopting units and individuals as secondary adopting units. This study will report the use and acceptance of systems development methodologies at the individual level. More specifically, it will study the use and acceptance of systems development methodologies among secondary adopters (individual system developers). The deployment of systems development methodologies among primary adopters (IS departments) is reported in Huisman and Iivari (2002).

Conceptual Research Model and Research Hypotheses

Theoretical Background

Most of the previous research into SDM did not have any theoretical orientation but the idea had been just to report the state of use of SDM and techniques in purely descriptive terms, e.g. Hardy et al. (1995) and Chatzoglou and Macaulay (1996). In general terms the present work is influenced by the diffusion of innovations (DOI) theory (Rogers, 1995), which is becoming an increasingly popular reference theory for empirical studies of information technologies (Beynon-Davies and Williams, 2003; Fichman, 1992; Prescott and Conger, 1995).

More specifically, this work is based on the IS implementation model suggested by Kwon and Zmud (1987). They combined IS implementation research and the DOI theory. This resulted in an enlarged model that identifies five categories of factors affecting IS implementation: individual factors, structural (organisational) factors, technological factors (innovation characteristics), task-related factors, and environmental factors. This categorisation provides the overall conceptual framework for this work (Figure 1). However, the selection of individual factors does not follow the model of Kwon and Zmud (1987) precisely for two reasons. Firstly, their list of 23 factors is quite comprehensive to be tested in one study. Secondly, the aim was to identify factors that are more specific to SDM than many of the factors they identified.

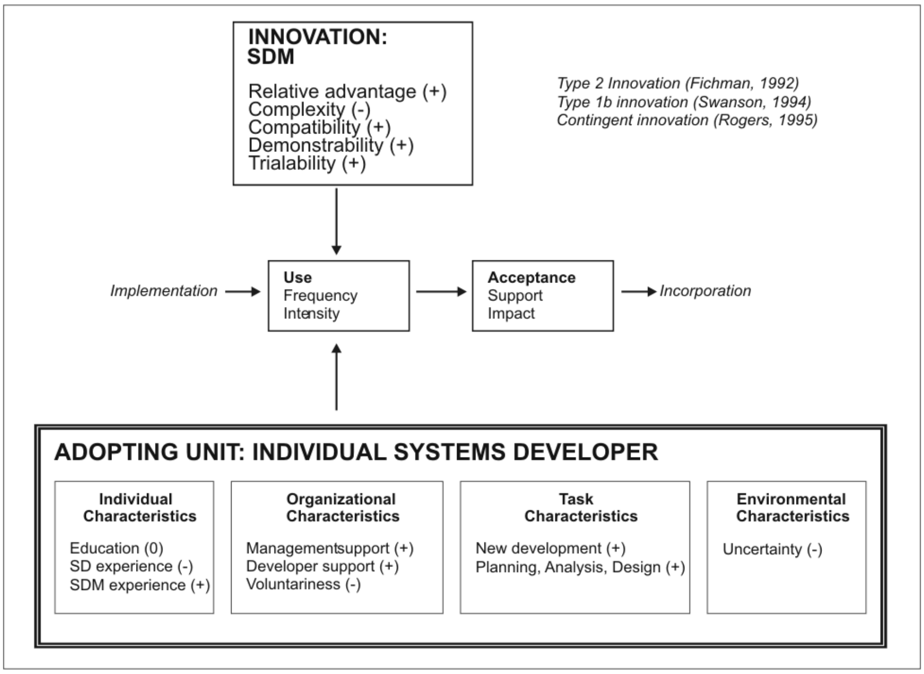

Fig. 1 Conceptual research model for the individual use and acceptance of SDM

According to the tri-core model presented by Swanson (1994), SDM are IS technological process innovations of Type 1b, which focus on the technical core of the IS department, and change the nature of IS work. The social system in this study is an organisation. In terms of Rogers (1995), SDM are contingent innovations with organisations as primary adopting units the individual level (Fichman, 1992; McChesney and Glass, 1993; Senn and Wynekoop, 1995; Dietrich *et al.,* 1997; Lai and Guynes, 1997). In terms of Rogers (1995), systems development methodologies are contingent innovations with organisations as primary adopting units and individuals as secondary adopting units. This study will report the use and acceptance of systems development methodologies at the individual level. More specifically, it will study the use and acceptance of systems development methodologies among secondary adopters (individual system developers). The deployment of systems development methodologies among primary adopters (IS departments) is reported in Huisman and Iivari (2002).

Conceptual Research Model and Research Hypotheses

Theoretical Background

Most of the previous research into SDM did not have any theoretical orientation but the idea had been just to report the state of use of SDM and techniques in purely descriptive terms, e.g. Hardy et al. (1995) and Chatzoglou and Macaulay (1996). In general terms the present work is influenced by the diffusion of innovations (DOI) theory (Rogers, 1995), which is becoming an increasingly popular reference theory for empirical studies of information technologies (Beynon-Davies and Williams, 2003; Fichman, 1992; Prescott and Conger, 1995).

More specifically, this work is based on the IS implementation model suggested by Kwon and Zmud (1987). They combined IS implementation research and the DOI theory. This resulted in an enlarged model that identifies five categories of factors affecting IS implementation: individual factors, structural (organisational) factors, technological factors (innovation characteristics), task-related factors, and environmental factors. This categorisation provides the overall conceptual framework for this work (Figure 1). However, the selection of individual factors does not follow the model of Kwon and Zmud (1987) precisely for two reasons. Firstly, their list of 23 factors is quite comprehensive to be tested in one study. Secondly, the aim was to identify factors that are more specific to SDM than many of the factors they identified.

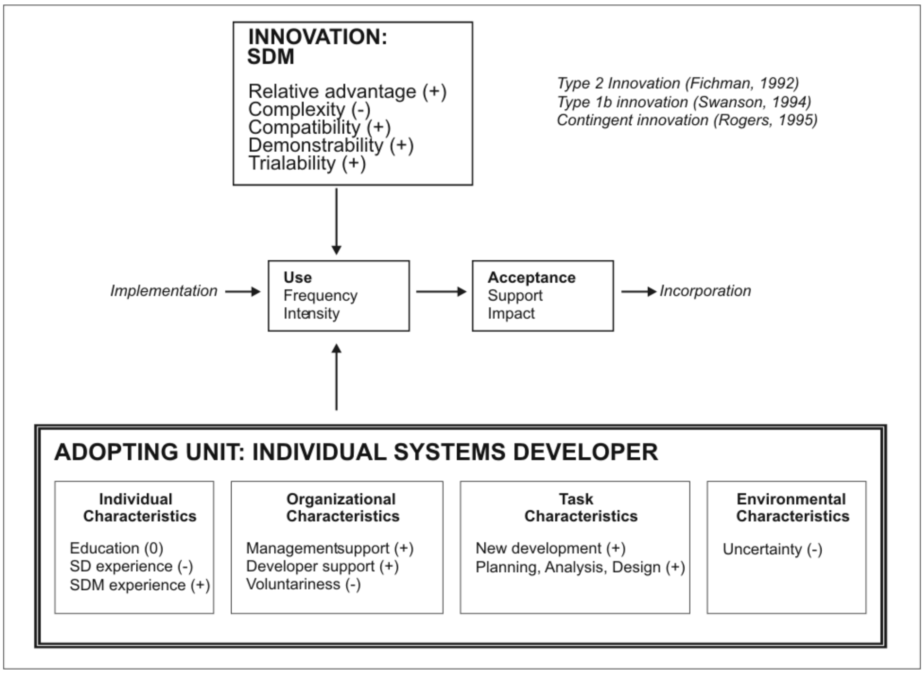

Fig. 1 Conceptual research model for the individual use and acceptance of SDM

According to the tri-core model presented by Swanson (1994), SDM are IS technological process innovations of Type 1b, which focus on the technical core of the IS department, and change the nature of IS work. The social system in this study is an organisation. In terms of

and individuals as secondary adopting units. This study will focus on the individual systems developer as the adopting unit.
The DOI theory has also been criticised. Fichman (1992) points out that it has mainly addressed individual adoption of relatively simple innovations. Despite of the author's recognition that SDMs are contingent innovations (Rogers, 1995), the focus in this paper lies on individual adoption of SDM. It is also obvious that SDM are fairly complex innovations. They are technologies of Type 2 (Fichman, 1992), which are characterised by a high knowledge burden or high user interdependencies. This means that this study tests the validity of DOI theory partly outside its major focus area. Therefore the detailed hypotheses concerning the deployment of SDM, derived from the classical DOI theory, are quite tentative.

As pointed above there is not much theoretically oriented empirical research into the adoption of SDM, on which to draw in the discussion of detailed hypotheses. To compensate this the author mainly uses existing empirical research on the adoption of CASE technology. There are two reasons for this. Firstly, CASE tools represent relatively complex technologies which are contingent innovations just as SDM. Secondly, the methodology companionship of CASE tools (Vessey et al., 1992) implies that their adoption includes a significant aspect of SDM.

The Innovation: SDM

Trying to define systems development methodologies is no easy task. There is no universally accepted, rigorous and concise definition of systems development methodologies (Avison and Fitzgerald, 1995; Wynekoop and Russo, 1997; Iivari et al., 1999). Avison and Fitzgerald (1995) argue that the term methodology is a wider concept than the term method, as it has certain characteristics that are not implied by method, i.e. the inclusion of a philosophical view. The author uses the term "methodology" to cover the totality of systems development approaches (e.g. structured approach, object-oriented approach), process models (e.g. linear life-cycle, spiral models), specific methods (e.g. IE, OMT, UML) and specific techniques.

Individuals' responses to an innovation are primarily influenced by attributes of the innovation and the implementation process (Leonard-Barton, 1987). The characteristics of SDM that will be studied was suggested by Rogers (1995), namely perceived relative advantage, compatibility, complexity, trialability and observability.

Relative advantage

Relative advantage is the degree to which an innovation is perceived as being better than the idea it supersedes (Rogers, 1995). After a decade's intensive research on TAM (Davis et al., 1989) in particular, there is significant empirical evidence that relative advantage or perceived usefulness (Moore and Benbasat, 1991) is positively related to innovation use, even though Iivari (1996) discovered it to have a significant relationship with CASE usage only at the organizational level but not at the individual level. This overwhelming evidence leads to the following hypothesis:

H1: There is a positive relationship between relative advantage and the individual use and acceptance of SDM.

Complexity

Complexity is the degree to which an innovation is perceived as difficult to understand and use (Rogers, 1995). It is generally believed that the more complex an individual perceives an innovation to be before using it, the less likely it is that the innovation will be adopted and implemented. Although perceived complexity has generally been assumed to be negatively related to the adoption of innovations (Davis *et al*., 1989; Moore and Benbasat, 1991; Rogers, 1995), the empirical results regarding the relationship between perceived ease of use (or perceived complexity) and use has been inconclusive (Gefen and Straub, 2000). This is also the case in the adoption of CASE tools (McChesney and Glass, 1993; Iivari, 1996).

Rogers (1995), SDM are contingent innovations with organisations as primary adopting units and individuals as secondary adopting units. This study will focus on the individual systems developer as the adopting unit.

The DOI theory has also been criticised. Fichman (1992) points out that it has mainly addressed individual adoption of relatively simple innovations. Despite of the author's recognition that SDMs are contingent innovations (Rogers, 1995), the focus in this paper lies on individual adoption of SDM. It is also obvious that SDM are fairly complex innovations. They are technologies of Type 2 (Fichman, 1992), which are characterised by a high knowledge burden or high user interdependencies. This means that this study tests the validity of DOI theory partly outside its major focus area. Therefore the detailed hypotheses concerning the deployment of SDM, derived from the classical DOI theory, are quite tentative.

As pointed above there is not much theoretically oriented empirical research into the adoption of SDM, on which to draw in the discussion of detailed hypotheses. To compensate this the author mainly uses existing empirical research on the adoption of CASE technology. There are two reasons for this. Firstly, CASE tools represent relatively complex technologies which are contingent innovations just as SDM. Secondly, the methodology companionship of CASE tools (Vessey et al., 1992) implies that their adoption includes a significant aspect of SDM.

The Innovation: SDM

Trying to define systems development methodologies is no easy task. There is no universally accepted, rigorous and concise definition of systems development methodologies (Avison and Fitzgerald, 1995; Wynekoop and Russo, 1997; Iivari et al., 1999). Avison and Fitzgerald (1995) argue that the term methodology is a wider concept than the term method, as it has certain characteristics that are not implied by method, i.e. the inclusion of a philosophical view. The author uses the term “methodology” to cover the totality of systems development approaches (e.g. structured approach, object-oriented approach), process models (e.g. linear life-cycle, spiral models), specific methods (e.g. IE, OMT, UML) and specific techniques.

Individuals' responses to an innovation are primarily influenced by attributes of the innovation and the implementation process (Leonard-Barton, 1987). The characteristics of SDM that will be studied was suggested by Rogers (1995), namely perceived relative advantage, compatibility, complexity, trialability and observability.

Relative advantage

Relative advantage is the degree to which an innovation is perceived as being better than the idea it supersedes (Rogers, 1995). After a decade's intensive research on TAM (Davis et al., 1989) in particular, there is significant empirical evidence that relative advantage or perceived usefulness (Moore and Benbasat, 1991) is positively related to innovation use, even though Iivari (1996) discovered it to have a significant relationship with CASE usage only at the organizational level but not at the individual level. This overwhelming evidence leads to the following hypothesis:

H1: There is a positive relationship between relative advantage and the individual use and acceptance of SDM.

Complexity

Complexity is the degree to which an innovation is perceived as difficult to understand and use (Rogers, 1995). It is generally believed that the more complex an individual perceives an innovation to be before using it, the less likely it is that the innovation will be adopted and implemented. Although perceived complexity has generally been assumed to be negatively related to the adoption of innovations (Davis *et al*., 1989; Moore and Benbasat, 1991; Rogers, 1995), the empirical results regarding the relationship between perceived ease of use (or perceived complexity) and use has been inconclusive (Gefen and Straub, 2000). This

Despite the inconclusive empirical evidence, the following is postulated in accordance with the DOI theory (Rogers, 1995) and TAM (Davis *et al.*, 1989):

H2: There is a negative relationship between complexity and the individual use and acceptance of SDM.

Compatibility

Compatibility is the degree to which an innovation is perceived as being consistent with the existing values, past experiences and needs of potential adopters, and it is positively related to innovation use (Rogers, 1995). Compatibility is sometimes described as the “fit” between an innovation and a particular context, which implies that an innovation must match its context in order to be effective. McChesney and Glass (1993) remark that a detailed assessment should be made of the “fit” between CASE methodology and the systems development tasks it is designed to support, when studying the acceptance of CASE methodology. Iivari (1996) also found some evidence for the significance of compatibility for CASE usage. Following the DOI theory, the next hypothesis is postulated as follows:

H3: There is a positive relationship between compatibility and the individual use and acceptance of SDM.

Demonstrability

Rogers (1995) uses the term “observability” and defines it as the degree to which the results of an innovation are visible to others. He argues that the easier it is for individuals to see the results of an innovation, the more likely they are to adopt it. Moore and Benbasat (1991) found that “observability” as originally defined by Rogers consist of two constructs, namely result demonstrability and visibility. Software-dominant innovations are less visible than hardware-dominant innovations (Moore and Benbasat, 1991; Rogers, 1995). Therefore the construct for result demonstrability is used and the following hypothesis is postulated as follows:

H4: There is a positive relationship between demonstrability and the individual use and acceptance of SDM.

Trialability

Rogers (1995) defines trialability as the degree to which an innovation may be experimented with on a limited basis. He argues that innovations, which are trialable, will be adopted more quickly, because it represents less uncertainty to the individual. The formulation of the fifth hypothesis is as follows:

H5: There is a positive relationship between trialability and the individual use and acceptance of SDM.

Innovation Diffusion Process

The main focus is on the use and acceptance of systems development methodologies, which is related to the implementation and confirmation stages of the innovation-decision process as described by Rogers (1995). Since use and acceptance are part of the post-implementation stage, the description of McChesney and Glass (1993) is used in the conceptual research model. After implementation, use and acceptance will follow, which in turn is followed by incorporation. (See Fig.1)

Use will be studied along two dimensions, namely frequency of use and intensity of use (McChesney and Glass, 1993). The acceptance of systems development methodologies will be studied from two perspectives, namely their impact on systems development and the perceived support it provides (McChesney and Glass, 1993). When studying the impact of systems development methodologies, this study will focus on their impact on both the developed system and the development process (Heineman *et al.*, 1994; Wynekoop and Russo, 1997). The support that systems development methodologies provide will be studied is also the case in the adoption of CASE tools (McChesney and Glass, 1993; Iivari, 1996). Despite the inconclusive empirical evidence, the following is postulated in accordance with the DOI theory (Rogers, 1995) and TAM (Davis *et al.*, 1989):

H2: There is a negative relationship between complexity and the individual use and acceptance of SDM.

Compatibility

Compatibility is the degree to which an innovation is perceived as being consistent with the existing values, past experiences and needs of potential adopters, and it is positively related to innovation use (Rogers, 1995). Compatibility is sometimes described as the “fit” between an innovation and a particular context, which implies that an innovation must match its context in order to be effective. McChesney and Glass (1993) remark that a detailed assessment should be made of the “fit” between CASE methodology and the systems development tasks it is designed to support, when studying the acceptance of CASE methodology. Iivari (1996) also found some evidence for the significance of compatibility for CASE usage. Following the DOI theory, the next hypothesis is postulated as follows:

H3: There is a positive relationship between compatibility and the individual use and acceptance of SDM.

Demonstrability

Rogers (1995) uses the term “observability” and defines it as the degree to which the results of an innovation are visible to others. He argues that the easier it is for individuals to see the results of an innovation, the more likely they are to adopt it. Moore and Benbasat (1991) found that “observability” as originally defined by Rogers consist of two constructs, namely result demonstrability and visibility. Software-dominant innovations are less visible than hardware-dominant innovations (Moore and Benbasat, 1991; Rogers, 1995). Therefore the construct for result demonstrability is used and the following hypothesis is postulated as follows:

H4: There is a positive relationship between demonstrability and the individual use and acceptance of SDM.

Trialability

Rogers (1995) defines trialability as the degree to which an innovation may be experimented with on a limited basis. He argues that innovations, which are trialable, will be adopted more quickly, because it represents less uncertainty to the individual. The formulation of the fifth hypothesis is as follows:

H5: There is a positive relationship between trialability and the individual use and acceptance of SDM.

Innovation Diffusion Process

The main focus is on the use and acceptance of systems development methodologies, which is related to the implementation and confirmation stages of the innovation-decision process as described by Rogers (1995). Since use and acceptance are part of the post-implementation stage, the description of McChesney and Glass (1993) is used in the conceptual research model. After implementation, use and acceptance will follow, which in turn is followed by incorporation. (See Fig.1)

Use will be studied along two dimensions, namely frequency of use and intensity of use (McChesney and Glass, 1993). The acceptance of systems development methodologies will be studied from two perspectives, namely their impact on systems development and the perceived support it provides (McChesney and Glass, 1993). When studying the impact of systems development methodologies, this study will focus on their impact on both the developed system and the development process (Heineman *et al.*, 1994; Wynekoop and along three dimensions, namely the perceived support as production technology, control technology, and cognitive/co-operative technology (Henderson and Cooprider, 1990).

Individual Characteristics

Education

Empirical results regarding the influence of education on the innovation process have been inconclusive. Kwon and Zmud (1987) state that negative associations have been found between education and innovation usage, mixed results have been reported for performance, and negative associations have been found between education and satisfaction with the innovation. This leads to the next hypothesis in its null form:

H6: There is no relationship between education and the individual use and acceptance of SDM.

Experience in systems development

Research suggests that the experience profile of an individual is an important factor in the acceptance and use of SDM. The findings suggest that inexperienced developers are more likely to use SDM and CASE tools (Lee and Kim, 1992; Orlikowski, 1993), while experienced developers may resist using them (Tesch et al., 1995). This leads to the following hypothesis:

H7: There is a negative relationship between an individual's experience in systems development and the individual use and acceptance of SDM.

Experience in SDM

Contrary to Hypotheses H7 Leonard-Barton (1987) argues that experienced developers are more likely to use SDM as they would be more aware of the benefits. Assuming that SDM have benefits, one can assume experienced developers who have experience of using a particular SDM to be more likely to use SDM. Therefore, the following hypothesis is postulated as follows:

H8: There is a positive relationship between an individual's experience with SDM and the individual use and acceptance of SDM.

Task Characteristics

SDM are primarily used in the development of new systems, and are often not as helpful in the enhancement of operational systems (Peacham, 1985). Furthermore, SDM are most effective for analysing the functionality of a newly developed system (Isoda et al., 1995). This leads to the following hypotheses:

H9: There is a positive relationship between the time an individual spends on the development of new systems and the individual use and acceptance of SDM.

H10: There is a positive relationship between the time an individual spends on the planning, analysis and design of a new system and the individual use and acceptance of SDM.

Organisational Characteristics

Individual systems developers do not work and deploy SDM in vacuum but under various social influences. The Theory of Reasoned Action (Fishbein and Ajzen, 1975) posits that an individual's behavior is influenced by his/her perception of the social pressure (subjective norm) to perform or not to perform the behavior. Subjective norm is defined as a multiplicative function of his/her normative beliefs (i.e. perceived expectations of specific referent individual and groups) and his/her motivation to comply with these expectations. The relevant referent groups may be top managers, supervisors, peers and friends. Karahanna et al. (1999) tested the influence of a number of referent groups on the subjective norms of

Russo, 1997). The support that systems development methodologies provide will be studied along three dimensions, namely the perceived support as production technology, control technology, and cognitive/co-operative technology (Henderson and Cooprider, 1990).

Individual Characteristics

Education

Empirical results regarding the influence of education on the innovation process have been inconclusive. Kwon and Zmud (1987) state that negative associations have been found between education and innovation usage, mixed results have been reported for performance, and negative associations have been found between education and satisfaction with the innovation. This leads to the next hypothesis in its null form:

H6: There is no relationship between education and the individual use and acceptance of SDM.

Experience in systems development

Research suggests that the experience profile of an individual is an important factor in the acceptance and use of SDM. The findings suggest that inexperienced developers are more likely to use SDM and CASE tools (Lee and Kim, 1992; Orlikowski, 1993), while experienced developers may resist using them (Tesch et al., 1995). This leads to the following hypothesis:

H7: There is a negative relationship between an individual's experience in systems development and the individual use and acceptance of SDM.

Experience in SDM

Contrary to Hypotheses H7 Leonard-Barton (1987) argues that experienced developers are more likely to use SDM as they would be more aware of the benefits. Assuming that SDM have benefits, one can assume experienced developers who have experience of using a particular SDM to be more likely to use SDM. Therefore, the following hypothesis is postulated as follows:

H8: There is a positive relationship between an individual's experience with SDM and the individual use and acceptance of SDM.

Task Characteristics

SDM are primarily used in the development of new systems, and are often not as helpful in the enhancement of operational systems (Peacham, 1985). Furthermore, SDM are most effective for analysing the functionality of a newly developed system (Isoda et al., 1995). This leads to the following hypotheses:

H9: There is a positive relationship between the time an individual spends on the development of new systems and the individual use and acceptance of SDM.

H10: There is a positive relationship between the time an individual spends on the planning, analysis and design of a new system and the individual use and acceptance of SDM.

Organisational Characteristics

Individual systems developers do not work and deploy SDM in vacuum but under various social influences. The Theory of Reasoned Action (Fishbein and Ajzen, 1975) posits that an individual's behavior is influenced by his/her perception of the social pressure (subjective norm) to perform or not to perform the behavior. Subjective norm is defined as a multiplicative function of his/her normative beliefs (i.e. perceived expectations of specific referent individual and groups) and his/her motivation to comply with these expectations. The relevant referent groups may be top managers, supervisors, peers and friends. Karahanna et potential *adopters* and *users* of Microsoft Windows 3.1 software. They found that top management, supervisors and peers significantly underlie subjective norms for both groups, and additionally local computer specialists for users. On the other hand, they discovered subjective norms to have a significant influence on behavioral intention to adopt but not on behavioral intention to continue using. In line with Karahanna et al. (1999) this study will focus on manager influence, including both top management and IT managers, and peer developer influence.

Management Support

Management support is consistently reported to facilitate IS implementation (Ginzberg, 1981). When one considers SDM, Isoda et al. (1995), Roberts and Hughes (1996) and Roberts et al. (1998) list a lack of management commitment as one of the biggest obstacles to implementing SDM. Iivari (1996) also reports management support to have significant effects on CASE usage both at the individual and organizational level. This leads to the next hypothesis:

H11: There is a positive relationship between management support and the individual use and acceptance of SDM.

Developer support

To the author's knowledge there is not much previous research on peer influence on the use and acceptance of innovations in general, IT innovations more specifically, and SDM innovations in particular. Despite that the following hypothesis is proposed:

H12: There is a positive relationship between developer support for the systems development methodology and the individual use and acceptance of SDM.

Voluntariness

Voluntariness is the degree to which an innovation is perceived as being voluntary or of free will (Moore and Benbasat, 1991). When we consider contingent innovations, the secondary adopters rarely have complete autonomy regarding their adoption and use in the workplace. Furthermore, SDM are complex innovations, and unless management declares their use mandatory, systems developers will have difficulty to fit them into their tight schedule. Iivari (1996) found strong support for the negative influence of voluntariness on CASE usage. This leads to the next hypothesis:

H13: There is a negative relationship between voluntariness and the individual use and acceptance of SDM.

Environmental Characteristics

Uncertainty

The deployment of SDM can be justified as an investment in the maintainability of systems, supported by proper documentation and proper modular structuring. Therefore, if the future of the IS department is uncertain or under threat, it may decrease the motivation to deploy the SDM. Unfortunately, to the author's knowledge there is no previous work on this relationship. Rai and Howard (1994) concluded that users may be unwilling to support new initiatives in an IS department in which they have limited confidence. On the other hand, uncertainty is believed to stimulate innovation through an organisation's effort to survive and grow (Kwon and Zmud, 1987). On the whole, the next hypothesis is formulated as follows:

H14: There is a negative relationship between the uncertainty about the continued existence of an IS department and the individual use and acceptance of SDM.

al. (1999) tested the influence of a number of referent groups on the subjective norms of potential *adopters* and *users* of Microsoft Windows 3.1 software. They found that top management, supervisors and peers significantly underlie subjective norms for both groups, and additionally local computer specialists for users. On the other hand, they discovered subjective norms to have a significant influence on behavioral intention to adopt but not on behavioral intention to continue using. In line with Karahanna et al. (1999) this study will focus on manager influence, including both top management and IT managers, and peer developer influence.

Management Support

Management support is consistently reported to facilitate IS implementation (Ginzberg, 1981). When one considers SDM, Isoda et al. (1995), Roberts and Hughes (1996) and Roberts et al. (1998) list a lack of management commitment as one of the biggest obstacles to implementing SDM. Iivari (1996) also reports management support to have significant effects on CASE usage both at the individual and organizational level. This leads to the next hypothesis:

H11: There is a positive relationship between management support and the individual use and acceptance of SDM.

Developer support

To the author's knowledge there is not much previous research on peer influence on the use and acceptance of innovations in general, IT innovations more specifically, and SDM innovations in particular. Despite that the following hypothesis is proposed:

H12: There is a positive relationship between developer support for the systems development methodology and the individual use and acceptance of SDM.

Voluntariness

Voluntariness is the degree to which an innovation is perceived as being voluntary or of free will (Moore and Benbasat, 1991). When we consider contingent innovations, the secondary adopters rarely have complete autonomy regarding their adoption and use in the workplace. Furthermore, SDM are complex innovations, and unless management declares their use mandatory, systems developers will have difficulty to fit them into their tight schedule. Iivari (1996) found strong support for the negative influence of voluntariness on CASE usage. This leads to the next hypothesis:

H13: There is a negative relationship between voluntariness and the individual use and acceptance of SDM.

Environmental Characteristics

Uncertainty

The deployment of SDM can be justified as an investment in the maintainability of systems, supported by proper documentation and proper modular structuring. Therefore, if the future of the IS department is uncertain or under threat, it may decrease the motivation to deploy the SDM. Unfortunately, to the author's knowledge there is no previous work on this relationship. Rai and Howard (1994) concluded that users may be unwilling to support new initiatives in an IS department in which they have limited confidence. On the other hand, uncertainty is believed to stimulate innovation through an organisation's effort to survive and grow (Kwon and Zmud, 1987). On the whole, the next hypothesis is formulated as follows:

H14: There is a negative relationship between the uncertainty about the continued existence of an IS department and the individual use and acceptance of SDM.

RESEARCH DESIGN

The Survey

This study is part of a larger survey on the deployment of SDM in South Africa, which was conducted between July and October 1999. The 1999 IT Users Handbook (the most comprehensive reference guide to the IT industry in South Africa) was used and the 443 listed organizations were contacted via telephone to determine if they were willing to participate in the study. 213 organizations agreed to take part. A package of questionnaires was sent to a contact person in each organization who distributed it. This package consisted of one questionnaire to be answered by the IT manager, and a number of questionnaires to be answered by individual systems developers in the organization. The response rate of the survey was as follows: 83 organizations (39%), 234 developers (26%) and 73 managers (34%) responded. The responses came from organisations representing a variety of business areas, manufacturing (33%) and finance/banking/insurance (15%) as the major ones. At the individual level the respondents reported considerable experience in SD, 22% between 3 and 5 years, 23% between 5-19 years and 38% more than 10 years.

Measurement

Measurement of Dependent Variables

In Table 2 the measurement of the different aspects used to study the use and acceptance of SDM is depicted. Frequency of use was measured using a question of how frequently the respondent needed or applied SDM knowledge (never; once a month or less: a few times in a month; several times in a week; every working day). Intensity of use was measured as the maximum of the individual usage of 29 listed methods, possible other commercial methods and possible in-house developed methods.

Table 2: Measurement of use and acceptance

Deployment aspects	Perspectives	Measurement	Reliability
Use	Frequency of use	2 items	0.84
	Intensity of use		
Support	Production technology	11 items (see Iivari and Huisman, 2001)	0.94
	Control technology	9 items (see Iivari and Huisman, 2001)	0.94
	Cognitive/co-operation technology	11 items (see Iivari and Huisman, 2001)	0.91
Impact	Quality of developed systems	8 items (see Iivari and Huisman, 2001)	0.95
	Quality and productivity of development process	10 items (see Iivari and Huisman, 2001)	0.94

One might also ask what independent variables explain use and acceptance of SDM in total. To answer this question, factor analysis on the seven aspects of deployment was performed, and this resulted in one factor with a reliability of 0.82. To measure total individual use and acceptance, the factor scores resulting from the factor analysis was used.

Measurement of Independent Variables

The measurement of the independent variables is summarised in Table 3. As Table 3 shows most measures were adopted from earlier research, and they have high reliability. The two items of uncertainty concerned the threat that the IS department is disbanded and the uncertainty of the future of the IS department in the organization.

RESEARCH DESIGN

The Survey

This study is part of a larger survey on the deployment of SDM in South Africa, which was conducted between July and October 1999. The 1999 IT Users Handbook (the most comprehensive reference guide to the IT industry in South Africa) was used and the 443 listed organizations were contacted via telephone to determine if they were willing to participate in the study. 213 organizations agreed to take part. A package of questionnaires was sent to a contact person in each organization who distributed it. This package consisted of one questionnaire to be answered by the IT manager, and a number of questionnaires to be answered by individual systems developers in the organization. The response rate of the survey was as follows: 83 organizations (39%), 234 developers (26%) and 73 managers (34%) responded. The responses came from organisations representing a variety of business areas, manufacturing (33%) and finance/banking/insurance (15%) as the major ones. At the individual level the respondents reported considerable experience in SD, 22% between 3 and 5 years, 23% between 5-19 years and 38% more than 10 years.

Measurement

Measurement of Dependent Variables

In Table 2 the measurement of the different aspects used to study the use and acceptance of SDM is depicted. Frequency of use was measured using a question of how frequently the respondent needed or applied SDM knowledge (never; once a month or less: a few times in a month; several times in a week; every working day). Intensity of use was measured as the maximum of the individual usage of 29 listed methods, possible other commercial methods and possible in-house developed methods.

Table 2: Measurement of use and acceptance

Deployment aspects	Perspectives	Measurement	Reliability
Use	Frequency of use	2 items	0.84
	Intensity of use		
Support	Production technology	11 items (see Iivari and Huisman, 2001)	0.94
	Control technology	9 items (see Iivari and Huisman, 2001)	0.94
	Cognitive/co-operation technology	11 items (see Iivari and Huisman, 2001)	0.91
Impact	Quality of developed systems	8 items (see Iivari and Huisman, 2001)	0.95
	Quality and productivity of development process	10 items (see Iivari and Huisman, 2001)	0.94

One might also ask what independent variables explain use and acceptance of SDM in total. To answer this question, factor analysis on the seven aspects of deployment was performed, and this resulted in one factor with a reliability of 0.82. To measure total individual use and acceptance, the factor scores resulting from the factor analysis was used.

Measurement of Independent Variables

The measurement of the independent variables is summarised in Table 3. As Table 3 shows most measures were adopted from earlier research, and they have high reliability. The two items of uncertainty concerned the threat that the IS department is disbanded and the uncertainty of the future of the IS department in the organization.

Table 3: Measurement of independent variables

Type	Characteristic	Measurement	Reliability
Innovation Characteristics	Relative advantage	5 items (adapted from Moore and Benbasat (1991))	0.94
	Complexity	3 items (adapted from Moore and Benbasat (1991))	0.88
	Compatibility	3 items (adapted from Moore and Benbasat (1991))	0.91
	Demonstrability	3 items (adapted from Moore and Benbasat (1991))	0.85
	Trialability	2 items (adapted from Moore and Benbasat (1991))	0.70
Individual Characteristics	Education	Highest qualification obtained	-
	Experience in systems development	Number of years	-
	Experience in the use of SDM	Number of years	-
Task Characteristics	Time spent on the development of new applications	%	-
	Time spent on planning, analysis, and design activities	%	-
Organisational Characteristics	Management support	2 items (adapted from Iivari (1996))	0.86
	Developer support	1 item	-
	Voluntariness	2 items (adapted from Moore and Benbasat (1991))	0.82
Environmental Characteristic	Uncertainty	2 items	0.90

Data Analysis

Data analysis was performed on the developer responses using Statistica software. In the first analysis the seven different aspects of use and acceptance were treated separately as the dependent variables. To identify the most important independent variables that explain the dependent variables, best subset multiple regression analysis was performed. In a second analysis, total use and acceptance was treated as the dependent variable, and best subset multiple regression was performed.

Multiple regression analysis assumes interval or ratio scale measurement, linearity, homoscedasticity, i.e. the constancy of the residual terms across the values of the predictor variables, independence of residuals, normality of residuals, and no multicollinearity (Hair, 1992). These assumptions were tested and no violations were detected.

Table 3: Measurement of independent variables

Type	Characteristic	Measurement	Reliability
Innovation Characteristics	Relative advantage	5 items (adapted from Moore and Benbasat (1991))	0.94
	Complexity	3 items (adapted from Moore and Benbasat (1991))	0.88
	Compatibility	3 items (adapted from Moore and Benbasat (1991))	0.91
	Demonstrability	3 items (adapted from Moore and Benbasat (1991))	0.85
	Trialability	2 items (adapted from Moore and Benbasat (1991))	0.70
Individual Characteristics	Education	Highest qualification obtained	-
	Experience in systems development	Number of years	-
	Experience in the use of SDM	Number of years	-
Task Characteristics	Time spent on the development of new applications	%	-
	Time spent on planning, analysis, and design activities	%	-
Organisational Characteristics	Management support	2 items (adapted from Iivari (1996))	0.86
	Developer support	1 item	-
	Voluntariness	2 items (adapted from Moore and Benbasat (1991))	0.82
Environmental Characteristic	Uncertainty	2 items	0.90

Data Analysis

Data analysis was performed on the developer responses using Statistica software. In the first analysis the seven different aspects of use and acceptance were treated separately as the dependent variables. To identify the most important independent variables that explain the dependent variables, best subset multiple regression analysis was performed. In a second analysis, total use and acceptance was treated as the dependent variable, and best subset multiple regression was performed.

Multiple regression analysis assumes interval or ratio scale measurement, linearity, homoscedasticity, i.e. the constancy of the residual terms across the values of the predictor variables, independence of residuals, normality of residuals, and no multicollinearity (Hair, 1992). These assumptions were tested and no violations were detected.

RESULTS

The results of the best subset multiple regression analysis with the seven different aspects of use and acceptance of SDM as the dependent variables are presented in Table 4.

The last column of Table 4 contains the results of the regression analysis where total use and acceptance was the dependent variable. To confirm the results of the regression analysis with total use and acceptance as the dependent variable, canonical analysis was performed. Canonical analysis is used to determine the relationship between two sets of variables. The seven different aspects of use and acceptance formed the first set, and the fourteen independent variables the second set. The resulting Canonical R was 0.90 at the level of $p \leq 0{,}001$, and the eigenvalue of the first root was 0.81. The factor structures of the two sets confirmed the results of the factor analysis and the regression analysis.

Table 4: Results of the regression analysis

N=173	Frequency of use	Intensity of use	Support: Production technology	Support: Control technology	Support: Cognitive/ co-operation technology	Impact: System	Impact: Process	Total use and acceptance
Relative advantage	0.34**	0.24’	0.25**	0.45***	0.49***	0.17’	0.39***	0.39***
Complexity			0.09					
Compatibility	-0.15	-0.20	0.20*	0.11		0.44***	0.27**	0.17*
Demonstrability		0.15’				0.10’		0.06
Trialability	0.08	0.11	0.08	0.10’	0.12*		0.08	0.10*
Education			0.06				0.06	
SD experience	-0.17*		-0.12*			-0.12*	-0.09’	-0.11*
SDM experience	0.45***	0.21**	0.10		0.07			0.14**
Time: Develop new applications					0.10’		0.08	0.05
Time: Planning, Analysis, Design	0.23***							
Manager support	-0.16*		0.29***	0.25***	0.10	0.10	0.18**	0.16**
Developer support	0.10		0.11’		0.20**	0.08		0.11’
Uncertainty	-0.08		-0.13**	-0.15**		-0.07	-0.10’	-0.09*
Voluntariness	-0.13’		0.07				0.07	
R	0.71	0.40	0.82	0.75	0.77	0.76	0.78	0.86
R^2	0.51	0.16	0.68	0.57	0.60	0.58	0.60	0.74
Adjusted R^2	0.48	0.13	0.66	0.56	0.58	0.56	0.58	0.73
F	17.65***	6.60***	32.11***	45.28***	41.88***	33.39***	27.94***	46.43***

‘ $p \leq 0{,}10$ * $p \leq 0{,}05$ ** $p \leq 0{,}01$ *** $p \leq 0{,}001$

RESULTS

The results of the best subset multiple regression analysis with the seven different aspects of use and acceptance of SDM as the dependent variables are presented in Table 4.

The last column of Table 4 contains the results of the regression analysis where total use and acceptance was the dependent variable. To confirm the results of the regression analysis with total use and acceptance as the dependent variable, canonical analysis was performed. Canonical analysis is used to determine the relationship between two sets of variables. The seven different aspects of use and acceptance formed the first set, and the fourteen independent variables the second set. The resulting Canonical R was 0.90 at the level of $p \leq 0{,}001$, and the eigenvalue of the first root was 0.81. The factor structures of the two sets confirmed the results of the factor analysis and the regression analysis.

Table 4: Results of the regression analysis

N=173	Frequency of use	Intensity of use	Support: Production technology	Support: Control technology	Support: Cognitive/ co-operation technology	Impact: System	Impact: Process	Total use and acceptance
Relative advantage	0.34**	0.24’	0.25**	0.45***	0.49***	0.17’	0.39***	0.39***
Complexity			0.09					
Compatibility	-0.15	-0.20	0.20*	0.11		0.44***	0.27**	0.17*
Demonstrability		0.15’				0.10’		0.06
Trialability	0.08	0.11	0.08	0.10’	0.12*		0.08	0.10*
Education			0.06				0.06	
SD experience	-0.17*		-0.12*			-0.12*	-0.09’	-0.11*
SDM experience	0.45***	0.21**	0.10		0.07			0.14**
Time: Develop new applications					0.10’		0.08	0.05
Time: Planning, Analysis, Design	0.23***							
Manager support	-0.16*		0.29***	0.25***	0.10	0.10	0.18**	0.16**
Developer support	0.10		0.11’		0.20**	0.08		0.11’
Uncertainty	-0.08		-0.13**	-0.15**		-0.07	-0.10’	-0.09*
Voluntariness	-0.13’		0.07				0.07	
R	0.71	0.40	0.82	0.75	0.77	0.76	0.78	0.86
R^2	0.51	0.16	0.68	0.57	0.60	0.58	0.60	0.74
Adjusted R^2	0.48	0.13	0.66	0.56	0.58	0.56	0.58	0.73
F	17.65***	6.60***	32.11***	45.28***	41.88***	33.39***	27.94***	46.43***

‘ $p \leq 0{,}10$ * $p \leq 0{,}05$ ** $p \leq 0{,}01$ *** $p \leq 0{,}001$

DISCUSSION AND FINAL COMMENTS

In this paper the purpose was to study factors that influence the individual use and acceptance of SDM. Fourteen possible factors were identified and fourteen hypotheses were postulated about their relationship with the individual use and acceptance of SDM. These are summarised in Table 5.

The results show that the classical DOI theory (Rogers, 1995) is relevant and useful in the case of individual use and acceptance of complex innovations such as SDM (see Fichman (1992)). The results indicate that especially relative advantage, compatibility less systematically and trialability more weakly have significant positive relationships with the individual use and acceptance of SDM. Relative advantage is positively related to all seven different aspects of individual use and acceptance of SDM. This suggests that individual systems developers' decisions to use and accept SDM mainly take place on rational grounds. If a systems developer sees SDM to provide relative advantage he or she is prepared to use it and to derive the benefits of using it. While compatibilty is strongly related to the perceived impact of SDM on the developed system and the development process, it is perplexing that it has negative, although not significant, beta coefficients with the methodology use. One explanation may be that when a SDM is highly compatible with a developer's way of working, its use may be quite routine and even tacit. It may be that the items measuring methodology use were not fully able to capture this routine or tacit nature of SDM use. On the other hand, when a methodology is perceived to be compatible with one's way of working, its benefits are perceived to be higher in terms of its impact on the quality of the system to be developed and the development process. Although trialability is not significantly related to the many aspects of deployment, it is related to use and acceptance of SDM in total. On the other hand, contrary to the predictions of the DOI theory, complexity and demonstrability were not significantly related to the individual use and acceptance of SDM. In order to explain this findings, ANOVA/MANOVA analysis was performed to test whether there is a difference in the use and acceptance of SDM among developers that received formal or informal training on the one hand or no training on the other hand. Developers that received formal or informal training reported statistically significant higher values for al seven aspects of use and acceptance than developers who did not receive any training. Therefore one could argue that training decrease the complexity of SDM.

Table 5: Summary of results

H1	There is a positive relationship between relative advantage and the individual use and acceptance of SDM	Strongly supported
H2	There is a negative relationship between complexity and the individual use and acceptance of SDM	Not supported
H3	There is a positive relationship between compatibility and the individual use and acceptance of SDM	Partially supported
H4	There is a positive relationship between demonstrability and the individual use and acceptance of SDM	Not supported
H5	There is a positive relationship between trialability and the individual use and acceptance of SDM	Weakly supported
H6	There is <u>no</u> relationship between education and the individual use and acceptance of SDM	Supported
H7	There is a negative relationship between an individual's experience in systems development and the individual use and acceptance of SDM	Partially supported
H8	There is a positive relationship between an individual's experience with SDM and the individual use and acceptance of SDM	Partially supported
H9	There is a positive relationship between the time an individual spends on the development of new systems and the individual use and acceptance of SDM	Not supported

DISCUSSION AND FINAL COMMENTS

In this paper the purpose was to study factors that influence the individual use and acceptance of SDM. Fourteen possible factors were identified and fourteen hypotheses were postulated about their relationship with the individual use and acceptance of SDM. These are summarised in Table 5.

The results show that the classical DOI theory (Rogers, 1995) is relevant and useful in the case of individual use and acceptance of complex innovations such as SDM (see Fichman (1992)). The results indicate that especially relative advantage, compatibility less systematically and trialability more weakly have significant positive relationships with the individual use and acceptance of SDM. Relative advantage is positively related to all seven different aspects of individual use and acceptance of SDM. This suggests that individual systems developers' decisions to use and accept SDM mainly take place on rational grounds. If a systems developer sees SDM to provide relative advantage he or she is prepared to use it and to derive the benefits of using it. While compatibilty is strongly related to the perceived impact of SDM on the developed system and the development process, it is perplexing that it has negative, although not significant, beta coefficients with the methodology use. One explanation may be that when a SDM is highly compatible with a developer's way of working, its use may be quite routine and even tacit. It may be that the items measuring methodology use were not fully able to capture this routine or tacit nature of SDM use. On the other hand, when a methodology is perceived to be compatible with one's way of working, its benefits are perceived to be higher in terms of its impact on the quality of the system to be developed and the development process. Although trialability is not significantly related to the many aspects of deployment, it is related to use and acceptance of SDM in total. On the other hand, contrary to the predictions of the DOI theory, complexity and demonstrability were not significantly related to the individual use and acceptance of SDM. In order to explain this findings, ANOVA/MANOVA analysis was performed to test whether there is a difference in the use and acceptance of SDM among developers that received formal or informal training on the one hand or no training on the other hand. Developers that received formal or informal training reported statistically significant higher values for al seven aspects of use and acceptance than developers who did not receive any training. Therefore one could argue that training decrease the complexity of SDM.

Table 5: Summary of results

H1	There is a positive relationship between relative advantage and the individual use and acceptance of SDM	Strongly supported
H2	There is a negative relationship between complexity and the individual use and acceptance of SDM	Not supported
H3	There is a positive relationship between compatibility and the individual use and acceptance of SDM	Partially supported
H4	There is a positive relationship between demonstrability and the individual use and acceptance of SDM	Not supported
H5	There is a positive relationship between trialability and the individual use and acceptance of SDM	Weakly supported
H6	There is no relationship between education and the individual use and acceptance of SDM	Supported
H7	There is a negative relationship between an individual's experience in systems development and the individual use and acceptance of SDM	Partially supported
H8	There is a positive relationship between an individual's experience with SDM and the individual use and acceptance of SDM	Partially supported
H9	There is a positive relationship between the time an individual spends on the development of new systems and the individual use and acceptance of SDM	Not supported

H10	There is a positive relationship between the time an individual spends on the planning, analysis and design of a new system and the individual use and acceptance of SDM	Not supported
H11	There is a positive relationship between management support and the individual use and acceptance of SDM	Supported
H12	There is a positive relationship between developer support for the systems development methodology and the individual use and acceptance of SDM	Weakly supported
H13	There is a negative relationship between voluntariness and the individual use and acceptance of SDM	Not supported
H14	There is a negative relationship between the uncertainty about the continued existence of an IS department and the individual use and acceptance of SDM	Weakly supported

Systems development experience was negatively related to the individual use and acceptance of SDM. This is in accordance with earlier findings that methodologies are used more by beginners than experienced developers (Lee and Kim, 1992; Orlikowski, 1993). However, to complicate the situation, experience with SDM was positively related to the individual use and acceptance of SDM. More experienced systems developers had more experience with SDM as indicated by the correlation coefficient ($r = 0{,}47$, $p \leq 0{,}001$). So, as individuals they comprise two characteristics that are opposite to each other with regard to the deployment of SDM: experience with SDM being positively associated with the deployment and SD experience being negatively associated with the deployment. When one considers the different aspects of use and acceptance of SDM the picture becomes a bit clearer. SD experience is mainly negatively related to the perceived impact of SDM on the developed system and the development process, and SDM experience is strongly positively related to the use of SDM.

The results also lend support for the significance of social influences on the individual use and acceptance of SDM. A significant positive relationship was found between management support and the individual use and acceptance of SDM. Furthermore, an almost significant positive relationship was found between developer support and the individual use and acceptance of SDM. These results are in line with previous research on the adoption of complex innovations (Fichman, 1992) and confirm earlier findings on the significance of management support in the case of software process improvement initiatives (Herbsleb et al., 1997).

The above results have clear practical implications. Assuming that use and acceptance of SDM are desirable, one should attempt to ensure that the individual systems developers perceive the methodologies to have relative advantage and compatibility with their work. The potential benefits of a methodology and reasons for its introduction should be made clear and communicated to the systems developers. However, the author does not believe that these perceptions can be changed using unilateral communication, but the benefits and compatibility should be discussed openly with systems developers. The significance of compatibility suggests that one should also be prepared to adapt and customise a methodology to fit the organization and the project as far as it does not threat the central reasons of introducing a methodology.

The trialability of SDM was also found significant. This would suggest that one should pay attention to the way of introducing a methodology. It can take place in pilot projects or applying only part of a methodology (e.g. starting from requirements engineering). As always in pilot projects one should pay attention to their selection as well to the selection of the participants. The participants should be motivated. Although it is not tested in this study, it would also be useful if they could serve as opinion leaders later, if the methodology is to be diffused more widely in the organisation. Referring to the significance of experience, it is important to ensure that the pilot project is as pleasant an experience as possible. The pilot project may be introduced as an opportunity to learn a new methodology that enhances and possibly updates the expertise of the participants (as could be the case when introducing some OO methodology).

H10	There is a positive relationship between the time an individual spends on the planning, analysis and design of a new system and the individual use and acceptance of SDM	Not supported
H11	There is a positive relationship between management support and the individual use and acceptance of SDM	Supported
H12	There is a positive relationship between developer support for the systems development methodology and the individual use and acceptance of SDM	Weakly supported
H13	There is a negative relationship between voluntariness and the individual use and acceptance of SDM	Not supported
H14	There is a negative relationship between the uncertainty about the continued existence of an IS department and the individual use and acceptance of SDM	Weakly supported

Systems development experience was negatively related to the individual use and acceptance of SDM. This is in accordance with earlier findings that methodologies are used more by beginners than experienced developers (Lee and Kim, 1992; Orlikowski, 1993). However, to complicate the situation, experience with SDM was positively related to the individual use and acceptance of SDM. More experienced systems developers had more experience with SDM as indicated by the correlation coefficient ($r = 0{,}47$, $p \leq 0{,}001$). So, as individuals they comprise two characteristics that are opposite to each other with regard to the deployment of SDM: experience with SDM being positively associated with the deployment and SD experience being negatively associated with the deployment. When one considers the different aspects of use and acceptance of SDM the picture becomes a bit clearer. SD experience is mainly negatively related to the perceived impact of SDM on the developed system and the development process, and SDM experience is strongly positively related to the use of SDM.

The results also lend support for the significance of social influences on the individual use and acceptance of SDM. A significant positive relationship was found between management support and the individual use and acceptance of SDM. Furthermore, an almost significant positive relationship was found between developer support and the individual use and acceptance of SDM. These results are in line with previous research on the adoption of complex innovations (Fichman, 1992) and confirm earlier findings on the significance of management support in the case of software process improvement initiatives (Herbsleb et al., 1997).

The above results have clear practical implications. Assuming that use and acceptance of SDM are desirable, one should attempt to ensure that the individual systems developers perceive the methodologies to have relative advantage and compatibility with their work. The potential benefits of a methodology and reasons for its introduction should be made clear and communicated to the systems developers. However, the author does not believe that these perceptions can be changed using unilateral communication, but the benefits and compatibility should be discussed openly with systems developers. The significance of compatibility suggests that one should also be prepared to adapt and customise a methodology to fit the organization and the project as far as it does not threat the central reasons of introducing a methodology.

The trialability of SDM was also found significant. This would suggest that one should pay attention to the way of introducing a methodology. It can take place in pilot projects or applying only part of a methodology (e.g. starting from requirements engineering). As always in pilot projects one should pay attention to their selection as well to the selection of the participants. The participants should be motivated. Although it is not tested in this study, it would also be useful if they could serve as opinion leaders later, if the methodology is to be diffused more widely in the organisation. Referring to the significance of experience, it is important to ensure that the pilot project is as pleasant an experience as possible. The pilot project may be introduced as an opportunity to learn a new methodology that enhances and possibly updates the expertise of the participants (as could be the case when introducing some OO methodology).

It is also significant that management communicates its support for the SDM introduction and use. The results also suggest that uncertainty concerning the future of the IS department is detrimental to methodology use and acceptance. Therefore, if the IS department is not under real threat, one should attempt to decrease such uncertainty.

REFERENCES

Avison, D.E., Fitzgerald, G. (1995) *Information Systems Development: Methodologies, Techniques and Tools*, McGraw-Hill, Berkshire, England

Beynon-Davies, P., Williams M.D. (2003) The diffusion of information systems development methods, *Strategic Information Systems,* Vol. 12, pp 29-46

Chatzoglou, P.D., Macaullay, L.A. (1996) Requirements capture and IS methodologies, *Information Systems Journal*, Vol. 6, pp 209-225

Davis, F.D., Bagozzi, R. P., Warshaw, P.R. (1989) User acceptance of computer technology: A comparison of two theoretical models, *Management Science*, Vol. 35, No. 8, pp 982-1003

Dietrich, G.B., Walz, D.B., Wynekoop, J.L. (1997) The failure of SDT Diffusion: A Case for Mass Customization, *IEEE Transactions on Engineering Management*, Vol. 44, No. 4, pp 390-398

Fichman, R.G. (1992) Information Technology Diffusion: A review of empirical research, in DeGross, J.I., Becker, J.D., Elam, J.J. (eds.) *Proceedings of the Thirteenth International Conference on Information Systems*, Dallas, TX, pp 195-206

Fishbein, M., Ajzen, I. (1975) *Belief, Attitude, Intention and Behavior: An Introduction to Theory and Research*, Addison-Wesley, Reading, MA

Fitzgerald, B. (1996) Formalized SDM: a critical perspective, *Information Systems Journal*, Vol. 6, pp 3-23

Fitzgerald, B. (1998) An empirical investigation into the adoption of SDM, *Information and Management*, Vol. 34, pp 317-328

Fitzgerald, B., Russo, N.L., O'Kane, T. (2003) Software Development Method Tailoring at Motorola, *Communications of the ACM,* Vol. 46, No. 4, pp 65-70

Gefen, D., Straub, D.W. (2000) The relative importance of perceived ease of use in IS adoption: A study of e-commerce adoption, *Journal of the Association for Information Systems*, Vol. 1, 2000

Ginzberg, M.J. (1981) Key recurrent issues in the MIS implementation process, *MIS Quarterly*, Vol. 5, pp 47-59

Hair, J.F., Anderson, R.E., Tatham, R.L., Black, W.C. (1992), *Multivariate Data Analysis with Readings*, Macmillan, New York

Hardy, C.J., Thompson J.B., Edwards H.M. (1995) The use, limitations and customization of structured systems development methods in the United Kingdom, *Information and Software Technology*, Vol. 37, No. 9, pp 467-477

Heineman, G.T., Botsford, J.E., Caldiera, G., Kaiser, G.E., Kellner, M.I., Madhavji, N.H. (1994) Emerging technologies that support a software process life cycle, *IBM Systems Journal*, Vol. 33, No. 3, pp 501-529

Henderson, J.C., Cooprider, J.G. (1990) Dimensions of I/S planning and design aids: A functional model of CASE technology, *Information Systems Research*, Vol. 1, No. 3, pp 227-254

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W., Paulk, M. (1997) Software quality and the Capability Maturity Model, *Communications of the ACM*, Vol. 40, No. 6, pp 30-40

It is also significant that management communicates its support for the SDM introduction and use. The results also suggest that uncertainty concerning the future of the IS department is detrimental to methodology use and acceptance. Therefore, if the IS department is not under real threat, one should attempt to decrease such uncertainty.

REFERENCES

Avison, D.E., Fitzgerald, G. (1995) *Information Systems Development: Methodologies, Techniques and Tools*, McGraw-Hill, Berkshire, England

Beynon-Davies, P., Williams M.D. (2003) The diffusion of information systems development methods, *Strategic Information Systems,* Vol. 12, pp 29-46

Chatzoglou, P.D., Macaullay, L.A. (1996) Requirements capture and IS methodologies, *Information Systems Journal*, Vol. 6, pp 209-225

Davis, F.D., Bagozzi, R. P., Warshaw, P.R. (1989) User acceptance of computer technology: A comparison of two theoretical models, *Management Science*, Vol. 35, No. 8, pp 982-1003

Dietrich, G.B., Walz, D.B., Wynekoop, J.L. (1997) The failure of SDT Diffusion: A Case for Mass Customization, *IEEE Transactions on Engineering Management*, Vol. 44, No. 4, pp 390-398

Fichman, R.G. (1992) Information Technology Diffusion: A review of empirical research, in DeGross, J.I., Becker, J.D., Elam, J.J. (eds.) *Proceedings of the Thirteenth International Conference on Information Systems*, Dallas, TX, pp 195-206

Fishbein, M., Ajzen, I. (1975) *Belief, Attitude, Intention and Behavior: An Introduction to Theory and Research*, Addison-Wesley, Reading, MA

Fitzgerald, B. (1996) Formalized SDM: a critical perspective, *Information Systems Journal*, Vol. 6, pp 3-23

Fitzgerald, B. (1998) An empirical investigation into the adoption of SDM, *Information and Management*, Vol. 34, pp 317-328

Fitzgerald, B., Russo, N.L., O'Kane, T. (2003) Software Development Method Tailoring at Motorola, *Communications of the ACM,* Vol. 46, No. 4, pp 65-70

Gefen, D., Straub, D.W. (2000) The relative importance of perceived ease of use in IS adoption: A study of e-commerce adoption, *Journal of the Association for Information Systems*, Vol. 1, 2000

Ginzberg, M.J. (1981) Key recurrent issues in the MIS implementation process, *MIS Quarterly*, Vol. 5, pp 47-59

Hair, J.F., Anderson, R.E., Tatham, R.L., Black, W.C. (1992), *Multivariate Data Analysis with Readings*, Macmillan, New York

Hardy, C.J., Thompson J.B., Edwards H.M. (1995) The use, limitations and customization of structured systems development methods in the United Kingdom, *Information and Software Technology*, Vol. 37, No. 9, pp 467-477

Heineman, G.T., Botsford, J.E., Caldiera, G., Kaiser, G.E., Kellner, M.I., Madhavji, N.H. (1994) Emerging technologies that support a software process life cycle, *IBM Systems Journal*, Vol. 33, No. 3, pp 501-529

Henderson, J.C., Cooprider, J.G. (1990) Dimensions of I/S planning and design aids: A functional model of CASE technology, *Information Systems Research*, Vol. 1, No. 3, pp 227-254

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W., Paulk, M. (1997) Software quality and the Capability Maturity Model, *Communications of the ACM*, Vol. 40, No. 6, pp 30-40

Huisman, H.M. and Iivari, J. (2002), “The organisational deployment of systems development methodologies”, in *Information Systems Development: Advances in Methodologies, Components, and Management,* Janis Grundspenski et al. (eds), Kluwer Press, ISBN 0-306-47698-3, pp 87-100, 2002

Huisman, H.M. and Iivari, J. (2003), “Reasons why systems development methodologies are not used”, in Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics, 27-30 July 2003, Orlando, Florida, Volume VI, Information Systems, Technologies and Applications: I, Nagib Callaos, William Lesso, Belkis Sánches, Elizabeth Hansen (eds), International Institute of Informatics and Systematics, ISBN 980-6560-01-9, pp 57-61.

Huisman, H.M. and Iivari, J. (2003), “Adaptation and the deployment of systems development methodologies”, in Proceedings of the International Conference on Computer, Communication and Control Technologies and the 9th International Conference on Information Systems Analysis and Synthesis, 31 July – 2 August 2003, Orlando, Florida, Volume I, Computing/Information Systems and Technologies, Hsing-Wei Chu, José Ferrer, María Sánchez, Edgar Acuña (eds), International Institute of Informatics and Systematics, ISBN 980-6560-05-1, pp 374-378.

Iivari J. (1996) Why are CASE Tools not used? *Communications of the ACM*, Vol.39, No.10, pp 94-103

Iivari, J., Hirscheim, R., Klein, H.K. (1999) Beyond Methodologies: Keeping up with Information Systems Development Approaches through Dynamic Classification, *Proceedings of the 32nd, Hawaian International Conference on Systems Sciences*, pp1-10

Iivari, J. and Huisman, M., The relationship between organisational culture and the deployment of systems development methodologies, in Dittrich, K., Geppert, A. and Norrie, M.C., *Advanced Information Systems Engineering*, Springer-Verlag, Berlin, 2001, pp 234-250

Introna, L.D., Whitley, E.A. (1997) Against methodism, Exploring the limits of methods, *Information Technology and People*, Vol.10, No. 1, pp 31-45

Isoda, S., Yamamoto, S., Kuroki, H., Oka, A. (1995) Evaluation and Introduction of the Structured Methodology and a CASE Tool, *Journal of Systems Software*, Vol.28, No.1, pp 49-58

Jayaratna, N. (1994) Understanding and Evaluation Methodologies, NIMSAD: A Systematic Framework, McGraw-Hill, London

Karahanna, E., Straub, D.W., Chervany, N.L. (1999) Information technology adoption across time: A cross-sectional comparison of pre-adoption and post-adoption beliefs, *MIS Quarterly*, Vol. 23, No. 2, pp 183-213

Kwon, T.H., Zmud, R.W. (1987) Unifying the Fragmented Models of Information Systems Implementation, in Boland, R.J., Hirschheim, R.A. (eds.) *Critical Issues in Information Systems Research*, John Wiley and Sons, New York, pp 227-251

Lai, V.S., and Guynes, J.L. (1997) An assessment of the influence of organizational characteristics on Information Technology Adoption Decision: A discriminative approach, *IEEE Transactions on Engineering Management,* Vol. 44, No. 2, pp146-157

Lee, J., Kim, S. (1992) The relationship between procedural formalization and MIS success, *Information and Management*, Vol.22, pp 89-111

Leonard-Barton, D. (1987) Implementing structured software methodologies: a case of innovation in process technology, *Interfaces*, Vol. 17, No. 3, pp 6-17

McChesney, I.R., Glass, D. (1993) Post-implementation management of CASE methodology, *European Journal of Information Systems*, Vol. 2, No. 3, pp 201-209

Huisman, H.M. and Iivari, J. (2002), “The organisational deployment of systems development methodologies”, in *Information Systems Development: Advances in Methodologies, Components, and Management,* Janis Grundspenski et al. (eds), Kluwer Press, ISBN 0-306-47698-3, pp 87-100, 2002

Huisman, H.M. and Iivari, J. (2003), “Reasons why systems development methodologies are not used”, in Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics, 27-30 July 2003, Orlando, Florida, Volume VI, Information Systems, Technologies and Applications: I, Nagib Callaos, William Lesso, Belkis Sánches, Elizabeth Hansen (eds), International Institute of Informatics and Systematics, ISBN 980-6560-01-9, pp 57-61.

Huisman, H.M. and Iivari, J. (2003), “Adaptation and the deployment of systems development methodologies”, in Proceedings of the International Conference on Computer, Communication and Control Technologies and the 9th International Conference on Information Systems Analysis and Synthesis, 31 July – 2 August 2003, Orlando, Florida, Volume I, Computing/Information Systems and Technologies, Hsing-Wei Chu, José Ferrer, María Sánchez, Edgar Acuña (eds), International Institute of Informatics and Systematics, ISBN 980-6560-05-1, pp 374-378.

Iivari J. (1996) Why are CASE Tools not used? *Communications of the ACM*, Vol.39, No.10, pp 94-103

Iivari, J., Hirscheim, R., Klein, H.K. (1999) Beyond Methodologies: Keeping up with Information Systems Development Approaches through Dynamic Classification, *Proceedings of the 32nd, Hawaian International Conference on Systems Sciences*, pp1-10

Iivari, J. and Huisman, M., The relationship between organisational culture and the deployment of systems development methodologies, in Dittrich, K., Geppert, A. and Norrie, M.C., *Advanced Information Systems Engineering*, Springer-Verlag, Berlin, 2001, pp 234-250

Introna, L.D., Whitley, E.A. (1997) Against methodism, Exploring the limits of methods, *Information Technology and People*, Vol.10, No. 1, pp 31-45

Isoda, S., Yamamoto, S., Kuroki, H., Oka, A. (1995) Evaluation and Introduction of the Structured Methodology and a CASE Tool, *Journal of Systems Software*, Vol.28, No.1, pp 49-58

Jayaratna, N. (1994) Understanding and Evaluation Methodologies, NIMSAD: A Systematic Framework, McGraw-Hill, London

Karahanna, E., Straub, D.W., Chervany, N.L. (1999) Information technology adoption across time: A cross-sectional comparison of pre-adoption and post-adoption beliefs, *MIS Quarterly*, Vol. 23, No. 2, pp 183-213

Kwon, T.H., Zmud, R.W. (1987) Unifying the Fragmented Models of Information Systems Implementation, in Boland, R.J., Hirschheim, R.A. (eds.) *Critical Issues in Information Systems Research*, John Wiley and Sons, New York, pp 227-251

Lai, V.S., and Guynes, J.L. (1997) An assessment of the influence of organizational characteristics on Information Technology Adoption Decision: A discriminative approach, *IEEE Transactions on Engineering Management,* Vol. 44, No. 2, pp146-157

Lee, J., Kim, S. (1992) The relationship between procedural formalization and MIS success, *Information and Management*, Vol.22, pp 89-111

Leonard-Barton, D. (1987) Implementing structured software methodologies: a case of innovation in process technology, *Interfaces*, Vol. 17, No. 3, pp 6-17

McChesney, I.R., Glass, D. (1993) Post-implementation management of CASE methodology, *European Journal of Information Systems*, Vol. 2, No. 3, pp 201-209

Moore, G.C., Benbasat, I. (1991) Development of an instrument to measure the perceptions of adopting an Information Technology innovation, *Information Systems Research*, Vol. 2, No. 3, pp 192-222

Nandhakumar, J., Avison, D.E. (1999) The fiction of methodological development: A field study of information systems development, *Information Technology and People*, Vol. 12, No. 2, pp 176-191

Orlikowski, W.J. (1993) CASE Tools as Organizational change: Investigating Incremental and Radical changes in Systems Development, *MIS Quarterly*, Vol. 17, No. 3, pp 309-340

Peacham, D. (1985) Structured methods – ten questions you should ask, *Data Processing*, Vol. 27, No. 9, pp 28-30

Prescott, M.B., Conger, S.A. (1995) Information Technology Innovations: A Classification by IT Locus of Impact and Research Approach, *The DATABASE for Advances in Information Systems*, Vol. 26, No. 2/3, pp 20- 42

Rahim, M., Seyal, A.H., Raham, N. (1998) Use of software systems development methods. An empirical study in Brunei Darussalam, *Information and Software Technology,* Vol. 39, pp 949-963

Rai, A., Howard, G.S. (1994) Propagating CASE usage for Software Development: An Empirical Investigation of Key Organizational Correlates, *OMEGA*, Vol. 22, No. 2, pp 133-147

Roberts, T.L., Hughes, C.T. (1996) Obstacles to implementing a systems development methodology, *Journal of Systems Management*, Vol. 47, pp 36-40

Roberts, T.L., Gibson, M.L., Fields, K.T., Rainer, R.K. (1998) Factors that impact implementing a SDM, *IEEE Transactions on Software Engineering*, Vol. 24, No.8, pp 640-649

Rogers, E.M. (1995) *Diffusion of Innovations*, The Free Press, New York

Russo, N.L., Hightower, R. and Pearson J.M. (1996) The failure of methodologies to meet the needs of current development environments, in Jayaratna, N., and Fitzgerald, B., (eds.) *Lessons learned from the Use of Methodologies: Fourth Conference on Information Systems Methodologies,* pp 387-394

Saeki, M. (1998) A meta-model for method integration, *Information and Software Technology,* Vol. 39, pp 925-932

Senn, J.A., and Wynekoop, J.L. (1995) The other side of CASE implementation, *Information Systems Management,* Vol. 12, No. 4, pp 7-14

Swanson, E.B. (1994) Information Systems Innovation Among Organizations, *Management Science*, Vol. 40, No. 9, pp 1069-1092

Tesch, D.B., Klein, G., Sobol, M.G. (1995) Information System Professionals' Attitudes: Development Tools and Concepts, *Journal of Systems Software*, Vol. 28, No.1, pp 39-47

Vessey, I., Jarvenpaa, S.L., Tractinsky, N. (1992) Evaluation of vendor products: CASE tools and methodology companions, Communications of the ACM, Vol. 35, No. 4, pp 90-105

Wynekoop J.L, Russo N.L (1997) Studying SDM: an examination of research methods, Information Systems Journal, Vol. 7, pp 47-65

Moore, G.C., Benbasat, I. (1991) Development of an instrument to measure the perceptions of adopting an Information Technology innovation, *Information Systems Research*, Vol. 2, No. 3, pp 192-222

Nandhakumar, J., Avison, D.E. (1999) The fiction of methodological development: A field study of information systems development, *Information Technology and People*, Vol. 12, No. 2, pp 176-191

Orlikowski, W.J. (1993) CASE Tools as Organizational change: Investigating Incremental and Radical changes in Systems Development, *MIS Quarterly*, Vol. 17, No. 3, pp 309-340

Peacham, D. (1985) Structured methods – ten questions you should ask, *Data Processing*, Vol. 27, No. 9, pp 28-30

Prescott, M.B., Conger, S.A. (1995) Information Technology Innovations: A Classification by IT Locus of Impact and Research Approach, *The DATABASE for Advances in Information Systems*, Vol. 26, No. 2/3, pp 20- 42

Rahim, M., Seyal, A.H., Raham, N. (1998) Use of software systems development methods. An empirical study in Brunei Darussalam, *Information and Software Technology,* Vol. 39, pp 949-963

Rai, A., Howard, G.S. (1994) Propagating CASE usage for Software Development: An Empirical Investigation of Key Organizational Correlates, *OMEGA*, Vol. 22, No. 2, pp 133-147

Roberts, T.L., Hughes, C.T. (1996) Obstacles to implementing a systems development methodology, *Journal of Systems Management*, Vol. 47, pp 36-40

Roberts, T.L., Gibson, M.L., Fields, K.T., Rainer, R.K. (1998) Factors that impact implementing a SDM, *IEEE Transactions on Software Engineering*, Vol. 24, No.8, pp 640-649

Rogers, E.M. (1995) *Diffusion of Innovations*, The Free Press, New York

Russo, N.L., Hightower, R. and Pearson J.M. (1996) The failure of methodologies to meet the needs of current development environments, in Jayaratna, N., and Fitzgerald, B., (eds.) *Lessons learned from the Use of Methodologies: Fourth Conference on Information Systems Methodologies,* pp 387-394

Saeki, M. (1998) A meta-model for method integration, *Information and Software Technology,* Vol. 39, pp 925-932

Senn, J.A., and Wynekoop, J.L. (1995) The other side of CASE implementation, *Information Systems Management,* Vol. 12, No. 4, pp 7-14

Swanson, E.B. (1994) Information Systems Innovation Among Organizations, *Management Science*, Vol. 40, No. 9, pp 1069-1092

Tesch, D.B., Klein, G., Sobol, M.G. (1995) Information System Professionals' Attitudes: Development Tools and Concepts, *Journal of Systems Software*, Vol. 28, No.1, pp 39-47

Vessey, I., Jarvenpaa, S.L., Tractinsky, N. (1992) Evaluation of vendor products: CASE tools and methodology companions, Communications of the ACM, Vol. 35, No. 4, pp 90-105

Wynekoop J.L, Russo N.L (1997) Studying SDM: an examination of research methods, Information Systems Journal, Vol. 7, pp 47-65

COPYRIGHT

COPYRIGHT

BYLAE B

Extreme Programming

Introduction

Extreme Programming (XP) is a new object-oriented software design process suited for small to medium-sized teams. Like any other traditional software development process, XP is goal-driven with a focus on communication, collaboration and a simplistic approach to the development process (Munro, 2003; Lippert and Roock, 2001; Müller and Tichy, 2001). This simplistic approach is what makes XP different from other development methodologies and has also contributed to it receiving more attention in the past few years.

Traditional methodologies have always required detailed documentation of all the steps followed in the process of designing the software. Emphasis is placed on the separate stages of planning, designing, testing and implementing the system. Extreme Programming effectively throws all this out the window. The only documentation in XP is a set of index cards on which users and team members write “stories” describing the planned features of the system. Other than that, the code is the only documentation. The reason behind this is that detailed documentation often slows down developers, and more often than not it is neglected in any case in the traditional design methodologies.

XP breaks development into bite-size chunks and relies on daily face-to-face communication and lots of testing; the assumption that specifications will change is also built into the process. Projects are built in increments with just enough code written to implement the required functions, and everything is constantly evaluated to accomplish the desired result.

Another contrast with tradition is that XP does not have separate phases for design, testing and implementation. Instead, these three phases are done together, incrementally and iteratively. Also, XP is unsophisticated because no attention is paid to the extensibility of the software. This means that, only the features that are required are planned for and implemented. (Müller, Tichy; 2001)

The XP Lifecycle

The Lifecycle of an Extreme Programming Project typically looks like **Figure 1** below.

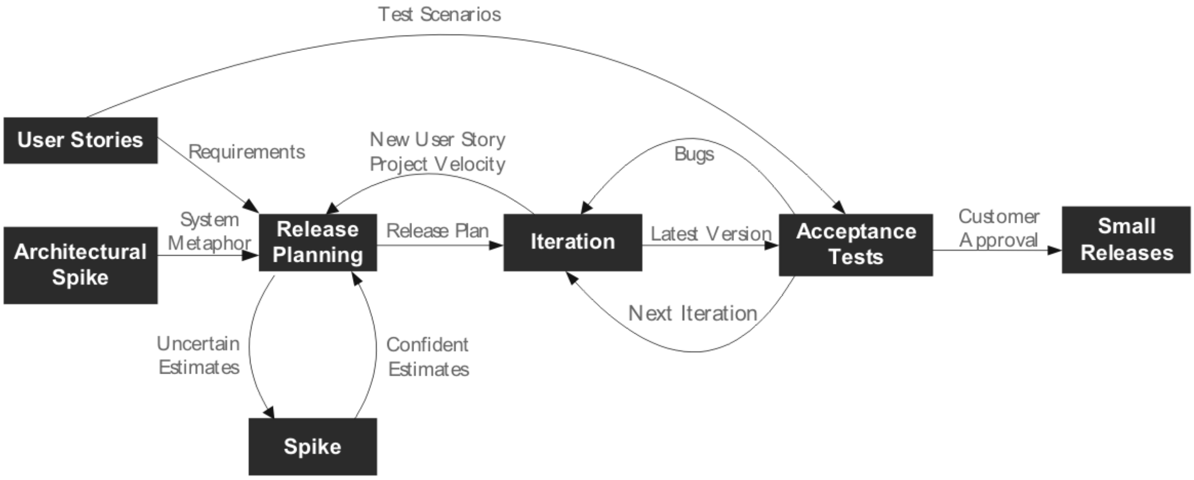

Figure 1: XP Lifecycle (Cohn and Paul, 2001)

ANNEXURE B

Extreme Programming

Introduction

Extreme Programming (XP) is a new object-oriented software design process suited for small to medium-sized teams. Like any other traditional software development process, XP is goal-driven with a focus on communication, collaboration and a simplistic approach to the development process (Munro, 2003; Lippert and Roock, 2001; Müller and Tichy, 2001). This simplistic approach is what makes XP different from other development methodologies and has also contributed to it receiving more attention in the past few years.

Traditional methodologies have always required detailed documentation of all the steps followed in the process of designing the software. Emphasis is placed on the separate stages of planning, designing, testing and implementing the system. Extreme Programming effectively throws all this out the window. The only documentation in XP is a set of index cards on which users and team members write “stories” describing the planned features of the system. Other than that, the code is the only documentation. The reason behind this is that detailed documentation often slows down developers, and more often than not it is neglected in any case in the traditional design methodologies.

XP breaks development into bite-size chunks and relies on daily face-to-face communication and lots of testing; the assumption that specifications will change is also built into the process. Projects are built in increments with just enough code written to implement the required functions, and everything is constantly evaluated to accomplish the desired result.

Another contrast with tradition is that XP does not have separate phases for design, testing and implementation. Instead, these three phases are done together, incrementally and iteratively. Also, XP is unsophisticated because no attention is paid to the extensibility of the software. This means that, only the features that are required are planned for and implemented. (Müller, Tichy; 2001)

The XP Lifecycle

The Lifecycle of an Extreme Programming Project typically looks like **Figure 1** below.

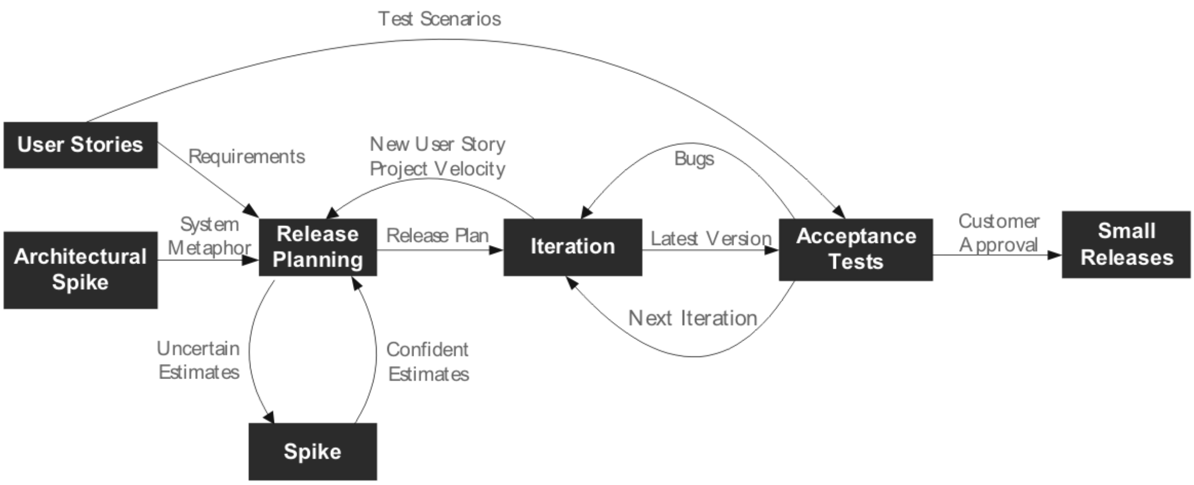

Figure 1: XP Lifecycle (Cohn and Paul, 2001)

The figure above encompasses the twelve core principles of XP which will be explained shortly. Refer to the figure to gain a better understanding of the Lifecycle and how the twelve core practices integrate into it.

The 12 Core Practices of Extreme Programming

Extreme Programming was created by Kent Beck in 1996 to fulfil a need for a faster, simpler and cheaper way to design software. In industry, organisations want to see results and see them fast; XP does just that and is based on twelve core practices

(Lippert and Roock, 2001; Munro, 2003):

- ***Pair Programming***

Two developers program together at one workstation. This increases the number of people that are familiar with the source code (Munro; Nuckols and Canna, 2003). The code has fewer bugs and it is easier to understand since there are two people reading the code at all times. Programmers switch seats periodically and regularly discuss each others code. This may seem like an ineffective way to program, but anecdotal evidence on the XP mailing list (extremeprogramming@yahoogroups.com) demonstrates success (English, 2002).

- ***Continuous Integration***

Small changes are frequently integrated into the common source base on a daily basis. Code is integrated and tested after every few hours or a day at the most. Integration of changes, one set at a time, simplifies the integration process and makes it obvious that who is responsible for fixing the code, when integration tests fail (English, 2002).

- ***Collective Ownership***

The code and all development documents are owned by the whole team. Any member is free to modify any part of the document at any time (English, 2002). This approach is very different to the traditional one where a single developer owns a set of code. XP supporters argue that the more people that work on a piece, the fewer bugs that will occur (English, 2002).

Waltman says that collective ownership plays a key part in meeting user's demands for system changes. “Last-minute changes can be a team effort, not an individual workload crisis”, he says.

- ***Small Releases***

The system is planned with short release cycles containing the most valuable business requirements. Typically, one cycle is less than three months and this allows the user to see and touch the working product on a regular basis (English, 2002).

- ***Testing***

Two types of tests are carried out continuously. Firstly, unit tests ensure that classes do what developers expect them to do. These tests are typically written by the developer (Munro; English, 2002).

Secondly, acceptance tests, written before the code that they will test, ensure that the system does what the user expects it to do. These tests are derived from the customer “stories” or scenarios (Munro; English, 2002; Nuckols and Canna, 2003).

All code has an associated test and new tests are added to old ones in a testing framework, creating a comprehensive test suite (Munro).

The figure above encompasses the twelve core principles of XP which will be explained shortly. Refer to the figure to gain a better understanding of the Lifecycle and how the twelve core practices integrate into it.

The 12 Core Practices of Extreme Programming

Extreme Programming was created by Kent Beck in 1996 to fulfil a need for a faster, simpler and cheaper way to design software. In industry, organisations want to see results and see them fast; XP does just that and is based on twelve core practices

(Lippert and Roock, 2001; Munro, 2003):

- ***Pair Programming***

Two developers program together at one workstation. This increases the number of people that are familiar with the source code (Munro; Nuckols and Canna, 2003). The code has fewer bugs and it is easier to understand since there are two people reading the code at all times. Programmers switch seats periodically and regularly discuss each others code. This may seem like an ineffective way to program, but anecdotal evidence on the XP mailing list (extremeprogramming@yahoogroups.com) demonstrates success (English, 2002).

- ***Continuous Integration***

Small changes are frequently integrated into the common source base on a daily basis. Code is integrated and tested after every few hours or a day at the most. Integration of changes, one set at a time, simplifies the integration process and makes it obvious that who is responsible for fixing the code, when integration tests fail (English, 2002).

- ***Collective Ownership***

The code and all development documents are owned by the whole team. Any member is free to modify any part of the document at any time (English, 2002). This approach is very different to the traditional one where a single developer owns a set of code. XP supporters argue that the more people that work on a piece, the fewer bugs that will occur (English, 2002).

Waltman says that collective ownership plays a key part in meeting user's demands for system changes. "Last-minute changes can be a team effort, not an individual workload crisis", he says.

- ***Small Releases***

The system is planned with short release cycles containing the most valuable business requirements. Typically, one cycle is less than three months and this allows the user to see and touch the working product on a regular basis (English, 2002).

- ***Testing***

Two types of tests are carried out continuously. Firstly, unit tests ensure that classes do what developers expect them to do. These tests are typically written by the developer (Munro; English, 2002).

Secondly, acceptance tests, written before the code that they will test, ensure that the system does what the user expects it to do. These tests are derived from the customer "stories" or scenarios (Munro; English, 2002; Nuckols and Canna, 2003).

All code has an associated test and new tests are added to old ones in a testing framework, creating a comprehensive test suite (Munro).

- ***Refactoring***

If a design is no longer appropriate, it is changed. This involves the restructuring of the software without changing its behaviour (Munro; English, 2002). This step, like many in XP, also requires user participation since it will be the user that will point out that a design no longer meets his/her needs. The combination of the test suite with refactoring, lets programmers avoid the classic "fix one thing, break another" bugs (Munro).

- ***Metaphor***

The system is based on a small set of powerful metaphors which guide the developers through the development process and explain the systems behaviour. These metaphors take the form of customer stories. The stories are high-level requirements of a system written in simple prose instead of the more traditional, and technical, Unified Modelling Language diagrams and models (Nuckols and Canna, 2003). Metaphors express the evolving project vision that defines the system's scope and purpose (English, 2002).

- ***Planning Game***

This stage in the development process encompasses requirements definition and project planning with releases being planned in the planning game (English, 2002). A release is broken into a number of 1-3 week iterations. The user basically lists the requirements of the system in this phase.

"Stories" describing these requirements are written by the user on index cards. The stories need not be in any particular order and are produced in a random manner. The programmers remove ambiguity by ensuring that they are estimable and testable.

The programmers also provide estimates on the stories so the user knows the cost of a particular story (Newkirk and Martin, 2000). Without a complex, wordy specification document, this low-tech technique leads to face-to-face communication between the user and the programming team. Keeping the written word to a minimum maintains discussion at a maximum (Munro, 2003)

The user then chooses which story will be implemented completely in the iteration and agrees not to make any changes or additions to it until the iteration completes. The user also needs to specify the completion criteria in the form of acceptance tests.

The programmers then break down the stories into tasks and determine the order in which they are to be implemented. Tasks are no longer than two days and the programmers sign up for the individual tasks and estimate the number of hours it will take to complete them based on personal experience. The tasks are summed and compared to the duration of the iteration. If there is too much to do the customer decides what to defer. If there is too little, the customer provides additional stories (Newkirk and Martin, 2000).

Thus the planning game is played again after an iteration and until the system is fully implemented.

- ***Coding Conventions***

A uniform style of the source code is used and this simplifies collective ownership. Thus all classes and methods are named in a standard way which is easily understood by the entire team, and a standard architecture is used to facilitate reuse of code (Munro). The programmers follow these rules so that all code looks as if it were written by a single person (English, 2002).

- ***Simple Design***

Simple designs are used because they are easy to create, maintain, understand and they discourage complexity. Design focuses on delivering a system that meets the user's

- ***Refactoring***

If a design is no longer appropriate, it is changed. This involves the restructuring of the software without changing its behaviour (Munro; English, 2002). This step, like many in XP, also requires user participation since it will be the user that will point out that a design no longer meets his/her needs. The combination of the test suite with refactoring, lets programmers avoid the classic “fix one thing, break another” bugs (Munro).

- ***Metaphor***

The system is based on a small set of powerful metaphors which guide the developers through the development process and explain the systems behaviour. These metaphors take the form of customer stories. The stories are high-level requirements of a system written in simple prose instead of the more traditional, and technical, Unified Modelling Language diagrams and models (Nuckols and Canna, 2003). Metaphors express the evolving project vision that defines the system’s scope and purpose (English, 2002).

- ***Planning Game***

This stage in the development process encompasses requirements definition and project planning with releases being planned in the planning game (English, 2002). A release is broken into a number of 1-3 week iterations. The user basically lists the requirements of the system in this phase.

“Stories” describing these requirements are written by the user on index cards. The stories need not be in any particular order and are produced in a random manner. The programmers remove ambiguity by ensuring that they are estimable and testable.

The programmers also provide estimates on the stories so the user knows the cost of a particular story (Newkirk and Martin, 2000). Without a complex, wordy specification document, this low-tech technique leads to face-to-face communication between the user and the programming team. Keeping the written word to a minimum maintains discussion at a maximum (Munro, 2003)

The user then chooses which story will be implemented completely in the iteration and agrees not to make any changes or additions to it until the iteration completes. The user also needs to specify the completion criteria in the form of acceptance tests.

The programmers then break down the stories into tasks and determine the order in which they are to be implemented. Tasks are no longer than two days and the programmers sign up for the individual tasks and estimate the number of hours it will take to complete them based on personal experience. The tasks are summed and compared to the duration of the iteration. If there is too much to do the customer decides what to defer. If there is too little, the customer provides additional stories (Newkirk and Martin, 2000).

Thus the planning game is played again after an iteration and until the system is fully implemented.

- ***Coding Conventions***

A uniform style of the source code is used and this simplifies collective ownership. Thus all classes and methods are named in a standard way which is easily understood by the entire team, and a standard architecture is used to facilitate reuse of code (Munro). The programmers follow these rules so that all code looks as if it were written by a single person (English, 2002).

- ***Simple Design***

Simple designs are used because they are easy to create, maintain, understand and they discourage complexity. Design focuses on delivering a system that meets the user’s immediate needs. No considerations are made for future functionality. The traditional method of building "hooks" for future functionality often leads to large systems with substantial elements that are initially unused and often discarded (English). In XP, if the design falls short of the required purpose, it is refactored.

- ***On-site Customer***

Since communication with the user is focused on in XP, contact with him/her is crucial. XP votes for a full-time customer on the development team. This ensures that the developers are always concentrating on the requirements and if they lose focus of these requirements, the customer is at hand to put them back on track (English, 2002).

- ***40 Hour Week***

Developing software the XP way is very demanding and developers need the weekends to rest, but as Beck says, "It's OK to work overtime when it is required, but don't do it two weeks in a row" (English, 2002). This assists in preventing burn-out in programmers and sloppy work that breaks down morale (Munro). Productivity is boosted when the developers are well-rested and motivated. In the long run this is favourable to the overall development of the system.

immediate needs. No considerations are made for future functionality. The traditional method of building “hooks” for future functionality often leads to large systems with substantial elements that are initially unused and often discarded (English). In XP, if the design falls short of the required purpose, it is refactored.

- ***On-site Customer***

Since communication with the user is focused on in XP, contact with him/her is crucial. XP votes for a full-time customer on the development team. This ensures that the developers are always concentrating on the requirements and if they lose focus of these requirements, the customer is at hand to put them back on track (English, 2002).

- ***40 Hour Week***

Developing software the XP way is very demanding and developers need the weekends to rest, but as Beck says, “It’s OK to work overtime when it is required, but don’t do it two weeks in a row” (English, 2002). This assists in preventing burn-out in programmers and sloppy work that breaks down morale (Munro). Productivity is boosted when the developers are well-rested and motivated. In the long run this is favourable to the overall development of the system.