Setup and Showcase of E-Commerce System

Naurin Jamil & Iarovyi Dmytro

TABLE OF CONTENTS

0 PREFACE

0.1 PURPOSE OF THIS DOCUMENT

This document is a Technical design document for use by the project. It provides guidance which is intended to assist the relevant management or technical staff, whether client or supplier, in understanding of the project. It is also useful background reading for anyone involved in further developing or monitoring the set-up and showcase of the e-commerce system,”E-Markt”.

0.2 USE OF THIS DOCUMENT

This document helps in providing better understanding of the technicalities of the project .Following is the explanation of the properties of the document.

a. “Summary” Properties

Title	Technical Design Document
Author	Naurin Jamil , Iarovyi Dmytro
Mentor	Dr. Fabrizio Orlandi

b. “Custom” Properties

Project Context	EIS Lab
Project Name	Set-up and showcase E-Commerce System
Group Id	Group K
Version	Issue 1
Date	09/30/14

1.3 BASIS OF THIS DOCUMENT

1. The following introductory sections set out an approach to setting up an E-commerce System. It attempts to analyse the different business processes, organizational models, data relationship diagrams through the modelling tool Aris. The use-cases have been designed using Astah and they help in simplifying the model design process. An important issue that needs to be considered is the availability of the model data online, so that it can be re-used by other consumers for extension and further usage.

2. A view points for designing the use-cases cover concerns for the online shop system, consumers interacting with it and the different outsourced service handling organizations. The Aris models help to translate these use cases in standard BPMN formats.

3. The concept of a converter is also introduced as part of the process for creating an interoperable environment which facilitates the exchange of information between the modelling tool and a web semantic format of RDF notations. These building blocks or components reflect the standard technologies that form the technical basis of the project development. These include the technologies that highlight model re-use, scalability and the creation of interoperable architectures around legacy environments.

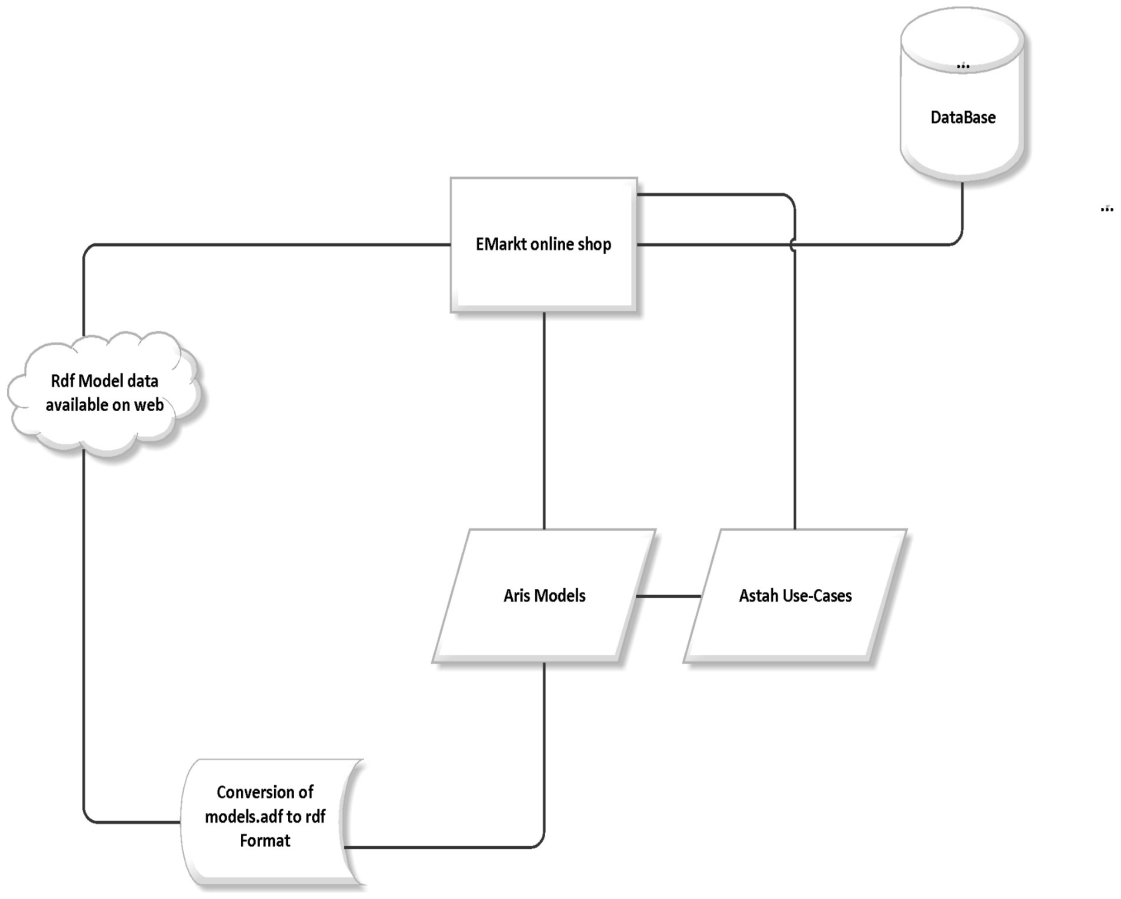
1.4 SYSTEM ARCHITECTURE FOR THE PROJECT

1. The different relevant use-cases for an online shop, "E-Markt" are designed using standard UML notations in Astah. These use-cases cover multiple scenarios like logging in the online shop, signing up, buying product, shipping product etc.

2. Aris modelling tool is then used to translate the use-cases for the different processes involved in the functioning of the e-commerce system to produce relevant models, like organizational chart, BPMN models, process landscape and the like. These models are now exported to XML file formats, running a script.

3. The Reference Aris models can now provide the input to a converter implementation that converts the XML format into the RDF notations. This makes it possible to publish the model data online to be consumed and re-used by others.

4.The Reference Architecture below would facilitate re-use throughout the project as individual models could be arranged to support the development of libraries of components that can be re-used. This will be one of the major potential benefits that could arise from the creation of the following Architecture.


DataBase
EMarkt online shop
Rdf Model data available on web
Aris Models
Astah Use-Cases
Conversion of models.adf to rdf Format

1.5 SPECIFIC DESIGN CONSIDERATIONS

1. Perhaps one of the most crucial aspects of this project is the designing of use-cases. They form the basis for all the models and extensions to semantic models. The use-cases must be drawn from the viewpoint of each stake holder involved in the working of "E-Markt". It could be the system admin, a consumer, a shipping company. Missing out any relevant use-case makes the modelling incomplete and a certain scenario gets missed out.

2. Aris modelling helps to translate the use-cases and provides many more features. It also helps to understand the internal organisational structure of the "E-Markt". It gives a better overview of the process landscape and how the process flows throughout the system. This gives a better understanding of the system to any internal or external user of the system. Modelling should be done precisely to match the use-cases exactly and concisely.

3. The models are used to generate XML files that retain model information in the most precise manner. This is made possible by running through a script that generates the attribute tags and corresponding values, for the models in a concise fashion. This script has to be implemented in such a way that it covers every aspect of the model.

4. Specific attention will also have to be placed in the choice of language for the implementation of the system. Today various options exist, and it is necessary as part of the design process to give very careful consideration to a range of trade-offs that need to be factored into development of a consistent design for the architecture extension. Languages such as Java, C, C++ and emerging languages, such as PYTHON, each offer different benefits to the developer. Careful consideration has to be given to the choice of which language to implement the system extension to gain the appropriate long-term benefits through the full life-cycle of the system.

5.This project has much to gain from adopting a standard approach to these projects based on, for example, the Unified Modelling Language (UML) paradigm, which is rapidly becoming recognised as an industry standard in software development. In the medium term this offers the exciting potential of creating classes or objects that can be agreed as standards to facilitate the creation of the interoperable architecture.

6. It is therefore recommended that the IDA Programme also standardise on a number of tools that support development and deployment of software in the projects associated with the programme. To implement systems based on UML, it is recommended that the standard BPMN modelling tools are only used. For this reason, Aris forms an appropriate choice.

1INTRODUCTION

1. This section should provide an overview of the entire document and a description of the scope of the system and its intended usage. The scope should also describe external interfaces to the system, external dependencies and provide a brief overview of the 'characteristics' of the system, commenting on aspects such as real-time use, security considerations, concurrency of users etc.

1.1PURPOSE

1. This document helps in comprehensive understanding of the e-commerce system that has been modelled as the online shop "E-Markt".

2. It has been written keeping in mind both the system developer/designer and system consumers in mind.

1.2SCOPE

1. a. The aspects dealt with in this system are the UML use cases for the different scenarios of the business processes; their corresponding Aris design models, their generated XML files and their corresponding RDF notations.

b. The project will help in better understanding as to what happens inside an operating e-commerce system and how the information flows to keep the system running.

c. This detailed description will be helpful to provide better understanding to a system designer or a consumer, also on the internet. It will make it easier to improve and extend functionalities as well.

d. As the information is made available online, there could be security risks involved with access of the modelling information.

e. Consistency is important for better understanding.

1.3DEFINITIONS, ACRONYMS AND ABBREVIATIONS

1. This section defines all terms, acronyms and abbreviations used in this document.

2. Terms covering the development of software using the Unified Modelling Language (UML) approach – Use-Case design

3. The following is a list of definitions based on the UML approach and ARIS models for system design:

Use Case Diagram -UML	Describes the system interactions with different stake holders
ARIS EPC	Event process chain diagrams to describe the different business processes
ARIS Organigram	Describes the organizational structure of the system
ARIS Process landscape	Gives an overview of the value-added processes in the system
General diagram	Design of the system architecture
BPMN	Enables modeling of processes using standard BPMN Notations

RDF	Resource Description Framework

1.4REFERENCES

1. This project references following publications:

Num.	Title (Applicability & Reference)	Author	Date	Issue
1	Dynamic Enterprise Architecture - From Static to Dynamic Models	Artur Latifov	Spring semester , 2012	06/14/12
2	The business process modeling ontology	Liliana Cabral , Barry Norton , John Domingue	2009	2009
3	Semantic EPC: Enhancing Process Modeling Using Ontology Languages	Oliver Thomas, Michael Fellmann	2007	04/07/07

1.5OVERVIEW

1. Section 1 is the introduction and includes a description of the project, applicable and reference documents.

2. Section 2 provides a system overview.

3. Section 3 contains the system context.

4. Section 4 describes the system design method, standards and conventions.

5. Section 5 contains the component descriptions.

6. Section 6 includes the Requirements Traceability Matrix.

2SYSTEM OVERVIEW

2.1SYSTEM CHARACTERISTICS

The description of the system:

▪system operates in real-time environment

▪the interfaces to view and design the UML models or Aris models are standard and user-friendly

▪a large number of viewpoints of the stake holders are considered

▪testing is done for model verification and code implementation

▪data publishing to ensure security

▪scalable and extensible model designs

▪data store to provide back up

2.2SYSTEM ARCHITECTURE

1. This section describes the Architecture of the system: The system under consideration is basically a modelling overview of an online-shop, taken as an example if E-Commerce system. The system is designed as a functional website that has a user interface as the front-end to interact with the users.

2. A database exists in the back-end that stores all the user related information and data about the products in the shop that are for sale and other warehouse related information, like products shipped, sold, unsold, damaged etc.

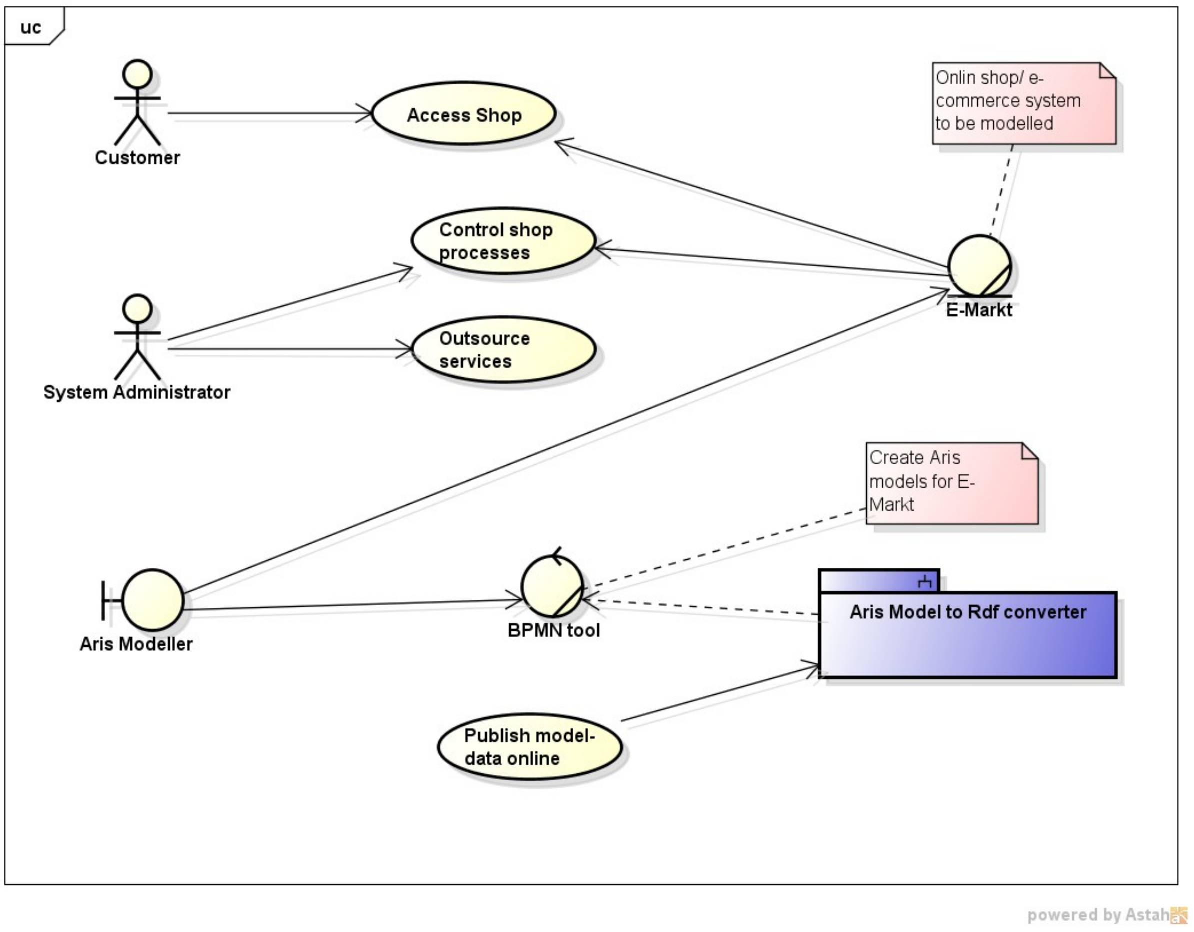
3. This system is taken up for building relevant use-cases, by a standard UML modelling tool that identifies the different scenarios that can arise while working with the system. Aris is chosen as an example tool.

4. Considering the different use-cases, it becomes easier to model the even process chains and business process models for the system, using Aris. It is also used to model the internal organisational structure of the system and process landscapes that provide a better overview of the system.

5. The system now exports data to an external converter that is designed to convert model information into the RDF format. This converter may be responsible to make the data available online for other users, to re-use or improve.

6. The system additionally interacts with external stake holders and business groups, to market products or outsources services like banking or logistics.

7. This illustrates the key point in the design of this system's architecture. The idea is not only to model processes but also to make this information usable and available to others and use the modern notations of semantic web.


uc
Customer
Access Shop
Control shop processes
Outsource services
System Administrator
Onlin shop/ e-commerce system to be modelled
E-Markt
Create Aris models for E-Markt
Aris Modeller
BPMN tool
Aris Model to Rdf converter
Publish model-data online
powered by Astah

2.3INFRASTRUCTURE SERVICES

1. Infrastructure Services should be provided to all applications with a view to reducing the time, cost and risks of development through re-use. To gain full advantage of infrastructure services the requirements of all current applications and the anticipated requirements from future applications should be analysed. These should be fed into the design of the services. The Infrastructure Services can be built incrementally, implementing the most common requirements first, followed by more specialised services. Infrastructure Services cover the following aspects:

a.Security

b.Audit and taxation

c.Performance monitoring and reporting

d.Error Handling

e.Debugging

f.Logging.

g.Persistent data store

h.Data re-use

i.Correctness and reliability

3SYSTEM CONTEXT

1. This section defines all the external interfaces and helps to illustrate the relationship between this system and other systems.

j.Information exchanged between system and its database is through standard sql queries. Other options of data access may also be considered.

k.The online transactions may use secure links, TCP/IP, especially for the banking transactions.

l.The information flow rate may be fast, along with the frequency of information being updated, the volume of information sent in a message block, how often the information is updated.

m.The system should have reliable interfaces that can deal with failover or communication link losses.

n.The emerging functionalities of cloud storage, web-semantics are utilised for system improvement. Other aspects of SOA, cloud computing may also be further considered.

o.Adherence to the standards wherever possible

4SYSTEM DESIGN

1. This and the following section should provide sufficient information for a developer to produce the system.

Code Description for **XML-RDF Converter:-**

a. Element_id - every element say Event, Function, organisational unit has its own element id

b. item_lists - for objects, connectors, models

c. graph - designed using the elements and specifying an ontology

d. model_lists - add each model from the list in the graph

e. object_item_lists - every object in each model, identified by its set of attributes, e.g. TypeNum

f. cxn_item_lists - every connector has its own target and source

g. after graph is created; it is serialized to output.txt file, in format n3

1. Amongst all the object design methodologies, UML is perhaps the best known and standardised.

2. The standards are also considered for designing the business process models (BPMN, EPC)

3. The XML format of the exported model data forms an extensible standard for the models

4. The RDF chosen for data publishing is itself emerging as a robust standard for the semantic web technology

4.1DESIGN METHOD AND STANDARDS

1. The design method is largely covered by modelling. The implementation occupies the extension features, described as below -

2. The script for the conversion of Aris models into XML is written in

3. The conversion of XML to RDF format follows standard mappings of the attributes in the models

4. The different notations in the EPC are: Events, Functions, Connectors, Organisational Unit IT System or Application, Error Event, Role

5. The different notations in use-case design are: Actors, Dependencies, Use-Cases, and Scenarios

4.2DOCUMENTATION STANDARDS

1. The guidelines used for the preparation of this document were taken from:

http://en.wikipedia.org/wiki/Technical_documentation

http://wiz.cath.vt.edu/tw/TechnicalWriting/docdesign/index.htm

4.3NAMING CONVENTIONS

1. This section explains all naming conventions, file formats used:

Adf – Aris models and designs

RDF – Resource Description Framework for data publishing

XML – extensible mark-up language, for intermediate conversion

4.4PROGRAMMING STANDARDS

1. XML to RDF converter standard –

2. RDF standard -

3. Aris models to XML standard -

4. In general, the programming standard should define a consistent and uniform programming style. Specific points to cover are:

p.modularity and structuring;

q.headers and commenting;

r.indenting and layout;

s.library routines to be used;

t.language constructs to use;

u.Language constructs to avoid.

4.5SOFTWARE DEVELOPMENT TOOLS

1. This section should list the tools chosen to assist software development, including testing. The actual software chosen will be heavily dependent upon the language in which the system will be implemented.

2. The list includes -

v.an application development tool;

w.a configuration manager / builder;

x.Testing tool - Aris

y.Microsoft word - a word processor for documentation;

z.Astah - a tool for drawing UML diagrams;

aa.Aris – a tool for drawing business process models bb.Automated testing tools.

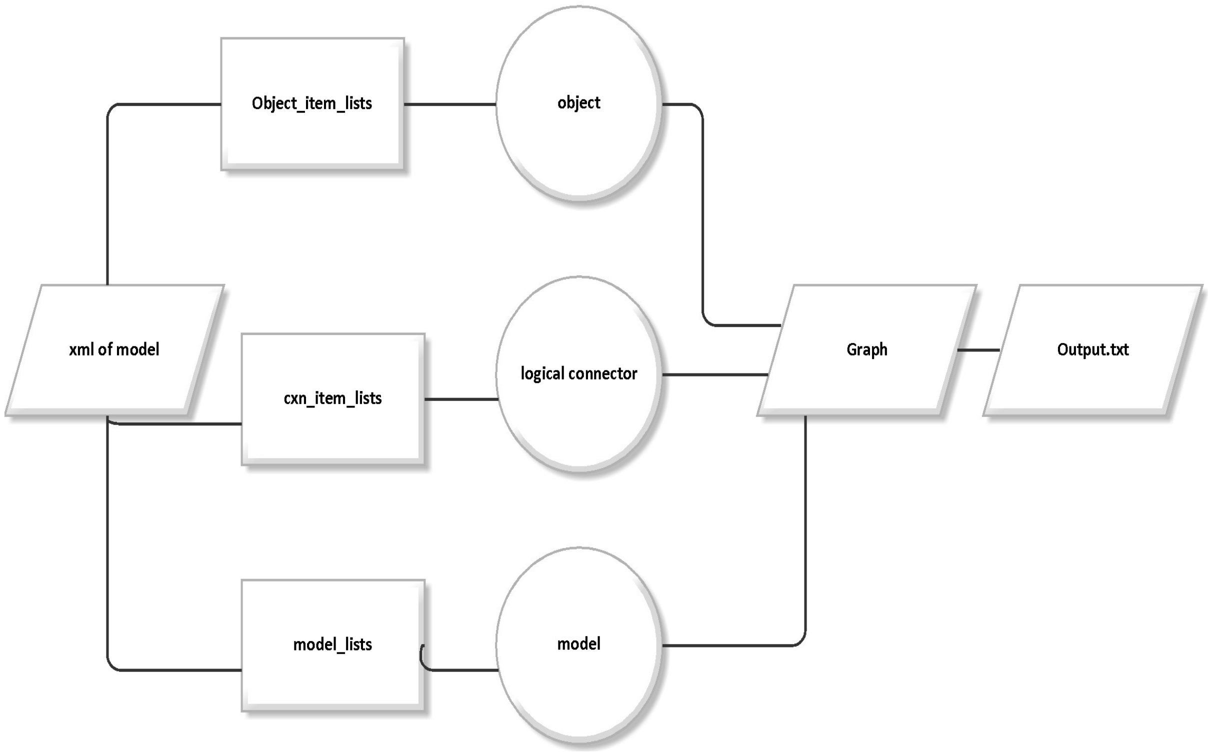
4.6OUTSTANDING ISSUES

1. No design issues exist.

4.7DECOMPOSITION DESCRIPTION

1. The software components are summarised. This should be presented as structure charts or object diagrams showing the hierarchy, control flow and data flow between the components.

2. If the UML paradigm is used then the decomposition description should make extensive use of the nine 'UML Diagrams' that in effect define the operation of the system.



5COMPONENT DESCRIPTION

For a software implementation, this and the previous section provides sufficient information for a programmer to produce the software, and for a maintainer, who may not be the developer, to make subsequent changes.

1. The descriptions of the components are laid out hierarchically.

▪5.1 Component identifier : **xml doc for the model**

▪5.1.1 Type : xml document

▪5.1.2 Purpose : capture model design in xml format

▪5.1.3 Function : data extension

▪5.1.5 Dependencies : model in Aris

▪5.1.6 Processing : written script

▪5.1.7 Data : design elements of model

▪5.2 Component identifier : **object_item_lists**

▪5.2.1 Type : list

▪5.2.2 Purpose : contain objects from the model

▪5.2.3 Function : container

▪5.2.4 Dependencies : XML doc for model in Aris

▪5.2.5 Processing : iteration

▪5.2.6 Data : all objects in the model design

▪5.3 Component identifier : **cxn_item_lists**

▪5.3.1 Type : list

▪5.3.2 Purpose : contain logical connectors from the model

▪5.3.3 Function : container

▪5.3.4 Dependencies : XML doc for model in Aris

▪5.3.5 Processing : iteration

▪5.3.6 Data : all logical connectors in the model design

▪5.4 Component identifier : **model_lists**

▪5.4.1 Type : list

▪5.4.2 Purpose : contain instances for the model

▪5.4.3 Function : container

▪5.4.4 Dependencies : XML doc for model in Aris

▪5.4.5 Processing : iteration

▪5.4.6 Data : all model instances in the model design

▪5.5 Component identifier : **Graph**

▪5.5.1 Type : graph

▪5.5.2 Purpose : represent XML data parsed as a graph

▪5.5.3 Function : parsing

▪5.5.4 Dependencies : lists

▪5.5.5 Processing : graph add methods

▪5.5.6 Data : graphical representation of parsed data

5.1COMPONENT IDENTIFIER

1. Each component should have a unique identifier. The identifiers to be used for components should be defined by the project and described elsewhere. For instance, XML doc for the model, object_item_lists, cxn_item_lists, model_lists, Graph

5.1.1Type

1. This section should describe the type of component, e.g. XML document, list/containers, graph, design etc

2.The contents of some component description sections depend on the component type. For the purpose of this template the categories: executable, i.e. contains computer instructions, or non-executable, i.e. contains only data, are used.

5.1.2Purpose

1.The purpose of a component should be defined by tracing it to the software requirements that it implements.

2.Backwards traceability depends upon each component description explicitly referencing the requirements that justify its existence.

5.1.3Function

1.The function of a component must be defined in this document. This should a short description of what the component does and will depend upon the component type e.g. it may be a description of the process or of the data to be stored or transmitted.

5.1.4Dependencies

1.The dependencies of a component should be defined by listing the constraints placed upon its use by other components. For example:

▪what operations have to have taken place before this component is called?

▪what operations are excluded when this operation is taking place?

▪what components have to be executed after this one?

5.1.5Processing

1.The processing should be defined by summarising the control and data flow within it. For some kinds of component, e.g. files, there is no such flow. Techniques of process specification include Program Design Language, Pseudo Code and Flow Charts.

2.Any specific algorithms to be used should be stated or referenced.

5.1.6Data

1.The data internal to a component should be defined. The amount of detail required depends strongly on the type of component. The logical and physical data structure of files that interface major components should be defined in detail.

2.Data structure definitions must include the:

a.description of each element, e.g. name, type, dimension;

b.relationships between the elements, i.e. the structure;

c.range of possible values of each element;

d.initial values of each element.

DOCUMENT CONTROL

Title:	Technical Design Document
Issue:	Issue 1
Date:	08 October, 2014
Author:	Naurin Jamil , Dymtro Iarovyi
Distribution:	Uni-Bonn
Reference:	IDA-MS-TD
Filename:	GroupK_Technical Design Document.odt
Control:	Reissue as complete document only

DOCUMENT SIGNOFF

Nature of Signoff	Person	Signature	Date	Role
Authors	Naurin Jamil , Dymtro Iarovyi			Project Member
Reviewers	Dr Fabrizio Orlandi			Project Mentor

DOCUMENT CHANGE RECORD

Date	Version	Author	Change Details
09/30/14	Issue 1 Draft 2	Naurin Jamil , Dymtro Iarovyi	First complete draft
10/02/14	Issue 1 Draft 3	Naurin Jamil , Dymtro Iarovyi	Review and update