Requirements Analysis

Contents

1 Preface 3

2 Introduction 3

2.1 IDMAPS background 3

2.2 Requirements Analysis 3

3 Process of developing the Requirements Analysis 4

3.1 The Data Audit 4

3.2 Systems and stakeholders 5

3.3 Data flows 5

4 General principles 5

4.1 Simplicity 5

4.2 Requirements prioritisation 5

5 Technical requirements 6

5.1 Data integrity checking 6

5.1.1 Field validation 6

5.1.2 Input accuracy screening 6

5.1.3 Feedback mechanism to master systems 6

5.2 Error reporting 6

5.2.1 Abnormal status reporting 6

5.2.2 Error notification system 7

5.3 Usage reporting and logging 7

5.3.1 Data analysis tools 7

5.3.2 Data load reporting 7

5.3.3 Statistical reporting of field quality 7

5.3.4 Final destination systems 7

5.3.5 Daily dashboard 7

5.3.6 Auditability and interrogability of end systems 8

5.4 Miscellaneous features 8

5.4.1 Grace periods 8

5.4.2 Mixed update frequency environment 8

5.4.3 Future-proofing (MOM, SOA, and Webservice capability) 8

5.4.4 Data update types 8
5.4.5 Redundancy and resilience 8
6 Procedural and policy requirements 9
6.1 Data auditing 9
6.2 Legal obligations 9
7 Project delivery requirements 10
7.1 Software tools 10
7.1.1 Collaborative documentation: MediaWiki 10
7.1.2 Version Control Repository: Subversion 10
8 Appendix A: systems and stakeholders list 11
9 Appendix B: data flow diagram 12

1 Preface

This document is the Requirements Analysis for the data infrastructure being developed by the IDMAPS project at Newcastle University.[1] It has been made available under a *Creative Commons Attribution-Share Alike 3.0 License* to the wider Higher Education community in the expectation that our experiences will prove useful to other institutions undertaking similar activities.[2]

Any references to third-party companies, products or services in this document are purely for informational purposes, and do not constitute any kind of endorsement by the IDMAPS Project or Newcastle University.

2 Introduction

2.1 IDMAPS background

Like any large institution, Newcastle University relies on a large amount and broad variety of institutional data in order to carry out its day-to-day operations. This data is stored across multiple computer systems, which have been implemented over a long period of time to address specific requirements (such as providing past exam papers, managing computer user accounts and student records, and providing Virtual Learning Environments).

These systems and their related data flows have grown organically, with little coherent planning for future expansion and development. This has led to several problems:

- Newly introduced systems often used new methods of importing the data they required from other systems, leading to numerous poorly documented, non-standardised and potentially unreliable data flows.
- Multiple copies of the same data often exist in several places in a variety of formats, without a known canonical source or adequate feedback mechanisms for correcting errors.
- A lack of standardised policies and procedures for obtaining and auditing access to institutional data served to further confuse the situation, and had the potential to lead to Freedom of Information and Data Protection failures.

IDMAPS aims to address these issues, improving the flow of institutional data within Newcastle University by implementing a logical, structured and extensible information architecture and supporting policies.

2.2 Requirements Analysis

The purpose of this document is to clarify the requirements of the IDMAPS project. These include:

- **Technical requirements of the solution** (e.g. it must do ***x***)
- **Procedural requirements of the solution** (e.g. doing ***x*** requires ***y*** policy)

[1] *Institutional Data Management for Personalisation and Syndication* (IDMAPS) is a JISC-funded Institutional Innovation project which aims to improve the quality and reliability of institutional data flows. For more information, please visit the project website at http://research.ncl.ac.uk/idmaps.
[2] http://creativecommons.org/licenses/by-sa/3.0/.

In addition, it provides details of **project delivery tools**, which the project team will use as part of the process of delivering the project.

This document is intended as a base from which to create and assess potential solutions for the information architecture and its supporting policies, and as such does not provide detailed specifications. For instance, this document does not specify the platform upon which the solution will be based.

3 Process of developing the Requirements Analysis

A significant challenge faced by the IDMAPS project team was identifying the various systems which created, stored or consumed institutional data within the University, as well as the data flows between them. Many of these systems and flows involved a wide range of University stakeholders such as the Library, Examinations Office, and Academic Schools.

Obtaining a clear picture of the existing data flow situation was a pre-requisite to creating an improved system which addressed the shortcomings of the existing situation. This was achieved by conducting a Data Audit[3].

3.1 The Data Audit

A stakeholder engagement session was held early on during the audit phase to bring together individuals from across the University who had an interest in and/or responsibility for data systems. The event contributed to the project team's understanding of the current data situation.

Having obtained an overview of systems and data flows, one-on-one meetings were arranged with system developers and owners. A member of the IDMAPS project team assisted these individuals to document data flows, including the production of data flow diagrams, for the systems they maintained. These diagrams visualised the individual systems, their linkages, and the associated data flows in a level of detail sufficient for the project team to begin to consider the technical requirements that IDMAPS should deliver.

The diagrams which were created were then displayed publicly for all parties to comment upon, annotate, add more information and/or correct where necessary. Edits were then incorporated into the original files. This improved the quality of information which had been obtained. In addition, it increased the visibility of the project within the University and encouraged the active participation of stakeholders.

The audit produced a number of outputs, including:

- A list of systems and stakeholders.
- Details of institutional data flows to and from systems, including methods, formats, frequencies, special requirements etc.
- A data dictionary documenting the institutional data identified during the audit.

[3] The IDMAPS project team has produced a set of guidelines for conducting a data audit based on their experience at Newcastle University. The document is aimed at other institutions which might benefit from the experiences of the IDMAPS team, and provides more details on the audit process. It is available from the IDMAPS website at http://research.ncl.ac.uk/idmaps/resources.php.

This information was used to create a rough list of core requirements for the IDMAPS solution, from which the remainder of this document is derived.

3.2 Systems and stakeholders

A list of core systems and stakeholders is attached to this document as Appendix A. It is not intended to be an exhaustive survey, but provide an overview of the most relevant systems and stakeholders to the project.

3.3 Data flows

A typical diagrammatic representation of data flows is attached to this document as Appendix B.

4 General principles

4.1 Simplicity

A key outcome of the IDMAPS project is the development and implementation of a new institutional data infrastructure. Whatever precise technical solution is decided upon and implemented, it should be as simple as possible whilst still meeting the essential requirements.

This is likely to improve the **reliability** of the data infrastructure, a key concern given critical nature of the dependant systems to the day-to-day operations of the University.

Furthermore, a simple solution can help to **mitigate security implications** by avoiding needless complexity.

It is worth noting that this principle of simplicity may prevent some desirable but non-essential features from being implemented, if they would significantly increase the complexity of the system.

For this reason, all proposed features should be clearly codified: only those essential to the data infrastructure should be labelled as such. The categorisation used is detailed below.

4.2 Requirements prioritisation

The IDMAPS project team have assigned one of the following three “importance” categorisations to every desired feature of the data infrastructure: **essential**, **desirable**, and **optional**. These words have the following definitions:

- An **essential requirement must** be delivered by the system for it to achieve its goals. An example of an essential requirement might be the ability to handle both interval-based data dumps and real-time updates, if there are examples of both types of system in essential use.
- A **desirable requirement** is one which we would like the system to offer. If the system provides it, we would normally expect to take advantage of this capability. However, if a desirable requirement is not met by a solution, the adoption of that solution would not be precluded.
- An **optional requirement** is one which is not significant for the project. If a solution can meet one or more optional requirements that is good, but such requirements are considered entirely optional.

Other institutions may assign different priorities to the features identified by the IDMAPS project team. Furthermore, other institutions may identify features specific to their institutional context which have not been identified by the IDMAPS project team at all.

5 Technical requirements

The IDMAPS project team identified a number of technical requirements of the data infrastructure

5.1 Data integrity checking

This consists of several individual requirements, each of which have been codified separately.

5.1.1 Field validation

Essential

The solution must ensure that every data field which is being passed to and from it is checked for integrity.

Incoming and outgoing data flows to different systems will have their own peculiar requirements. In addition, problems could arise with NULL fields, blank fields composed of spaces, the data type or length of fields. The solution must be able to deal with these cases gracefully, ensuring structural integrity.

5.1.2 Input accuracy screening

Optional

Ensure data is accurate at time of input (for instance, that an “age” field is not more than 150). This is not the same as validation, described above, and ensures syntactic integrity.

Although this feature would be useful, it is primarily the concern of the upstream systems rather than the data architecture.

5.1.3 Feedback mechanism to master systems

Essential

The solution must ensure that updates, corrections and deletions of data in systems downstream of the architecture are passed accurately and swiftly back to their respective upstream systems.

In addition, any such feedback made by the solution to master systems should be clearly logged.

Such feedback mechanism and associated logging will help to ensure semantic integrity.

5.2 Error reporting

5.2.1 Abnormal status reporting

Essential

The solution must report and log any abnormal statuses for the purposes of future reference/audit, as well as for fault diagnosis and rectification.

5.2.2 Error notification system

Essential

The solution must provide notification systems for any errors it encounters in the data it processes. This should be automated for scripts, and take the form of contact address for users and system administrators.

5.3 Usage reporting and logging

5.3.1 Data analysis tools

Desirable

It would be desirable if the solution could provide tools for analysing the data passing through it, such as Venn diagrams, charts, and tables. It would be desirable if this was easily exported (for example through .csv files) to other systems.

5.3.2 Data load reporting

Essential

The solution must provide details of the load under which the system is running. This should include information such as the quantities of data sent and received on a per-system and aggregate level.

5.3.3 Statistical reporting of field quality

Optional

If possible, the solution should provide statistical data on the quality of the fields it receives from other sources.

For instance, one might expect incoming data for a particular field to ordinarily consist of purely numeric values, but be unable to define it within the schema as a numeric field because other systems may provide alphanumeric output. In such circumstances, it may not be appropriate for the solution to simply discard such values as erroneous if they would not affect the day-to-day operation of the recipient system.

However, if the solution could provide statistical reporting on the extent of this kind of field quality issue, such information could assist in developing solutions to reducing the problem amongst source data systems in the first place.

5.3.4 Final destination systems

Essential

The solution must be able to maintain a record of which end-systems retrieve and use the data fields it creates.

5.3.5 Daily dashboard

Optional

If possible, the solution should provide a daily “dashboard” or similar summary of all activities.

5.3.6 Auditability and interrogability of end systems

Optional

If possible, the solution should be able to query the systems with which it interacts, such that it could be asked to display the source locations of data held on a particular resource or individual.

This would be particularly useful for audit purposes and error diagnosis.

5.4 Miscellaneous features

5.4.1 Grace periods

Optional

If possible, the solution should be able to handle "grace periods", where records might become inactive for a period of time but not be removed from systems. An example of this would be medical students transitioning between stages.

5.4.2 Mixed update frequency environment

Essential

The solution must be able to handle a mixed environment consisting of both overnight data dumps for certain legacy systems, and real-time updates for more modern systems.

5.4.3 Future-proofing (MOM, SOA, and Webservice capability)

Desirable

It would be desirable for the solution to support a messaging architecture. This would provide guaranteed delivery of information between systems, as well as allowing publish and subscribe capability when sending data to and from multiple systems.

The loose coupling of systems provided by a Service Orientated Architecture would be beneficial in helping to increase the resilience of the solution.

The solution should provide a set of web services for regular data-centric activity, easing day-to-day administration tasks.

5.4.4 Data update types

Desirable

It would be desirable if the solution could handle systems requiring both complete data reloads, as well as regular deltas of change.

5.4.5 Redundancy and resilience

Essential

The solution must provide redundancy and resilience, as it will be relied upon by numerous mission-critical University systems.

Precisely how this is achieved is open for discussion: in a decoupled message-based architecture, the temporary non-availability of a message queue will not cause the failure of the system. The alternatives would be delivering a solution with 24/7 uptime, or ensuring that all downstream and

upstream Applications are tolerant of downtime. Neither of these options is likely to be feasible, due to time and budget constraints.

6 Procedural and policy requirements

A fundamental problem with the current situation is that the policies and procedures which underpin existing data transfers not fully embedded.

This has significant potential consequences. Downstream systems have sometimes obtained their information from *other* downstream systems rather than from the source of the data, without regard for the potential inaccuracies and problems that depending on second- or third-hand information may cause. Furthermore, there are Freedom of Information and Data Protection considerations regarding the transfer of personal data between systems.

For these reasons, the solution implemented by the project **must** be fully supported by procedures and policies which ensure that data integrity is maintained, and that both data providers and consumers are aware of their legal and compliance obligations.

6.1 Data auditing

Essential

The solution must have policies which ensure that uses of data are audited.

Requests for data must be fully documented, and must include details of who made and authorised the request, the uses to which the data will be put, and how the data request will be met. The project will provide a Data Integration Template so that such requests are appropriately documented in a standardised form.[4]

Data feeds must be accompanied by usage agreements signed by both ISS and the Application Provider, which must include expiry conditions on the data provided.

6.2 Legal obligations

Essential

The solution must be clearly supported by policies which ensure that any legal or compliance obligations are met.

Data uses must comply with Newcastle University's existing policies which govern the collection, storage, use, and removal of data. Such policies include the Information Security Policy, the Data Protection Policy, and the Freedom of Information Policy.

In addition, any additional implications brought about by a data infrastructure which passes personal details between systems must be adequately addressed.

The creation of a data dictionary, to hold information about which systems possess and access which data, will assist the University in dealing more efficiently with DPA and FOI requests.

[4] The Data Integration Template, and Guidelines for Completing the Data Integration Template, are available from the IDMAPS project website at http://research.ncl.ac.uk/idmaps.

7 Project delivery requirements

Whilst considering the process of undertaking the IDMAPS project, the need for various other tools and requirements was identified. These are not requirements for the solution that the project will deliver.

7.1 Software tools

With numerous individuals from different departments involved in the IDMAPS project team, there is a need for software tools to manage the collation of information and data gathered and/or generated by the project.

7.1.1 Collaborative documentation: MediaWiki

The IDMAPS project team chose MediaWiki as a collaborative documentation tool.[5] This is used for drafting and revising documents, making notes (such as proposed content at planned events), and recording minutes and actions of meetings.

Using MediaWiki has numerous benefits. It allows the project team to work collaboratively and revert edits where necessary. It uses a lightweight markup syntax which is swift to learn, and therefore allows rapid development of documents. MediaWiki is robust, being well known as the software upon which Wikipedia is built, and has a strong user community behind it.

Information Systems and Services at Newcastle University has institutional knowledge of maintaining and deploying MediaWiki, as it already provides the software as a hosted service to institutional users.

7.1.2 Version Control Repository: Subversion

IDMAPS uses Subversion to power a shared data repository for the project.[6] This is used for storing files and data which have been generated or compiled over the course of the project, such as Visio diagrams of data flows.

Subversion was chosen as it is in widespread use, is well documented and stable, and is provided as a service by Information Systems and Services within Newcastle University.

[5] http://www.mediawiki.org/wiki/MediaWiki.
[6] http://subversion.tigris.org/.

8 Appendix A: systems and stakeholders list

A list of core systems and stakeholders identified during the data audit phase.

Name	Description	Purpose	Technical Description	Provider
Accommodation	Online accommodation system	Allows students to apply for University-owned accommodation	Black box	3rd Party, proprietary (Kinetics)
Active Directory	Windows directory service	Provides Windows accounts, IT asset management	Directory of user accounts and IT hardware	3rd Party, proprietary (Microsoft)
Blackboard	Virtual learning environment	Provides a virtual learning environment for students/staff	Java application, Tomcat	3rd Party, proprietary (Blackboard Inc.)
CAMA	Computer account management	Ties Active Directory and non-AD information together	.net application, Microsoft SQL database	Internal (ISS WIT)
Dspace	Scholarly archive system	Provides a repository for scholarly work	Java application, PostgreSQL database	3rd Party, open-source (DSpace Foundation)
ePortfolios	Portfolio-building system	Reflective learning tools and continuous professional development	Python applications, Zope, MySQL database	Internal (LTMS)
ePrints	Published scholarly work repository	Provides access to peer-reviewed work of Newcastle University staff	Perl application, MySQL database	Internal (Library)
Backtraq	Facilities management system	Allows Estates support issues to be raised, tracked and resolved	Web-based, Microsoft IIS	3rd Party, (Unknown)
Exam Papers	Online exam paper repository	Provides past exam papers for reference	PHP	Internal (ISS)
FMSC VLEs	Virtual learning environments	Provides a virtual learning environments for Medical students/staff	Python applications, Zope, MySQL database	Internal (FMSC)
Grouper	Group management system	Provides grouping control for user accounts across multiple systems	Java application, MySQL database	3rd Party, open-source (Internet2)
Individual Linker	Data linking system	Manages identities of individuals across multiple systems	.net application, Microsoft SQL database	Internal (DMS & BDD)
Intralibrary	Content repository	Provides a central medical content/resource storage, cataloguing and search system	Java application, MySQL database	Internal (LTMS & ISS)
MOFS (Module Outline Forms System)	Database containing module data	Provides users with a web application for maintaining the University's modules	.net application, Microsoft SQL database	Internal (ISS ADU)
MyImpact	Research management system	Allows monitoring of research impact for RAE submision	.net application, Microsoft SQL database	Internal (BDD)
MyProfiles	Staff profile management system	Staff personal web profiles, publications lists, personal plans and reseach activities	Python application, Zope, MySQL database	Internal (BDD & LTMS)
NESS	Virtual learning environment	Provides a virtual learning environment for Computer Science students/staff*	Perl application, MySQL database	Internal (SCS)
NUContacts	Tutor/Tutee management system	Allows tutors to display student data and contact their tutees	.net application, Microsoft SQL database	Internal (ISS ADU)
Print Credits	Printer credit system	Provides printer credit management and links to user accounts	Perl application, PostGreSQL database	Internal (ISS Unix)
Recap	Lecture capture system	Provides automated audio and video capture for lectures and events	Various (Lasso, MySQL database and others)	3rd Party, proprietary (Lectopia)
S3P (Student Self-Service Portal)	Online student details system	Allows students to register for academic modules and update personal details	SAP-based web-forms	Internal (ISS SAP)
ServiceCenter	Call management system	Allows ISS staff to manage and resolve IT incident reports raised by students/staff	Java application, Microsoft SQL database	3rd Party, proprietary (HP/Peregrine)
Shibboleth	Single sign-on management system	Provides single sign-on capabilities across multiple systems	Java application, MySQL database	3rd Party, open-source (Internet2)
SiteManager	Content management system	Provides CMS capabilities for some University websites	PHP application, MySQL database	Internal (ISS IAD)
Smartcard	Physical access control system	Allows fine- and coarse-grained access control to physical locations across campus	Black box, Microsoft SQL Server	3rd Party, proprietary (ESi, Chubb)
Student Homepage	Student news/information website	Displayed to all student users on log-on, providing student news and information	PHP application, MySQL database	Internal (CWD)
Syllabus Plus	Timetabling system	Provides automated timetabling and resource-management (e.g. room booking)	Black box, Microsoft SQL Server	3rd Party, proprietary (Scientia)
Unix	Various Unix services and systems	Underlying servers for SAP, Websites, transfer mechanism for other systems	Predominantly Linux, SAP on Solaris	Internal (ISS Unix)
www.ncl.ac.uk	University Website	Public University web presence	PHP application, MySQL database	Internal (CWD)
* Some other schools/faculties also use NESS				
Acronyms				
ISS	Information Systems and Services			
ADU	Application Development Unit			
LTMS	Learning Technologies for Medical Sciences			
WIT	Windows Infrastructure Team			
IAD	Information Applications and Delivery			
SAP	SAP team			
CWD	Corporate Web Development			
SCS	School of Computer Science			

9 Appendix B: data flow diagram